Guide du développeur

Amazon Elastic Container Service

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Elastic Container Service: Guide du développeur

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Les marques et les présentations commerciales d'Amazon ne peuvent être utilisées en relation avec un produit ou un service extérieur à Amazon, d'une manière susceptible de créer une confusion chez les clients, ou d'une manière qui dénigre ou discrédite Amazon. Toutes les autres marques commerciales qui ne sont pas la propriété d’Amazon appartiennent à leurs propriétaires respectifs, qui peuvent ou non être affiliés ou connectés à Amazon, ou sponsorisés par Amazon.

Table of Contents

Qu'est-ce que Amazon ECS ? ... 1
Terminologie et composants ... 1
Fonctionnalités ... 3
Allouer ... 3
Tarification ... 4
Services connexes ... 4
Premiers pas ... 6
Configuration ... 6
AWS Management Console ... 6
Inscrivez-vous pour un Compte AWS ... 7
Création d'un utilisateur doté d'un accès administratif ... 7
Créer un Virtual Private Cloud ... 9
Création d'un groupe de sécurité ... 10
Créez les informations d'identification pour vous connecter à votre EC2 instance ... 14
Installez le AWS CLI ... 14
Étapes suivantes pour utiliser Amazon ECS ... 15
Création d'une image de conteneur ... 16
Prérequis ... 16
Créer une image Docker ... 18
Transmettre votre image à Amazon Elastic Container Registry ... 20
Nettoyage ... 22
Étapes suivantes ... 22
Découvrez comment créer une tâche Linux pour le type de lancement Fargate ... 22
Prérequis ... 23
Étape 1 : créer le cluster ... 23
Étape 2 : Créer une définition de tâche ... 24
Étape 3 : Créer le service ... 26
Étape 4 : Afficher votre service ... 26
Étape 5 : nettoyer ... 26
Découvrez comment créer une tâche Windows pour le type de lancement Fargate ... 27
Prérequis ... 27
Étape 1 : créer un cluster ... 28
Étape 2 : Enregistrer une définition de tâche Windows ... 29
Étape 3 : Créer un service avec votre définition de tâche ... 30

Étape 4 : Afficher votre service ... 31
Étape 5 : Nettoyer ... 32
Découvrez comment créer une tâche Windows pour le type de EC2 lancement ... 32
Prérequis ... 32
Étape 1 : créer un cluster ... 33
Étape 2 : Enregistrer une définition de tâche ... 35
Étape 3 : Créer un service ... 36
Étape 4 : Afficher votre service ... 37
Étape 5 : Nettoyer ... 38
En utilisant le AWS CDK ... 39
Étape 1 : Configurez votre AWS CDK projet ... 40
Étape 2 : utilisez le AWS CDK pour définir un serveur Web conteneurisé sur Fargate ... 42
Étape 3 : Tester le service web ... 50
Étape 4 : nettoyer ... 50
Étapes suivantes ... 50
Création de ressources à l'aide de la AWS CLI Copilot ... 51
Installation de la CLI AWS Copilot ... 52
Déploiement d'un exemple d'application Amazon ECS à l'aide de la CLI AWS Copilot ... 61
Amazon ECS avec AWS CloudFormation ... 63
Exemples de AWS CloudFormation modèles ... 64
Définitions de tâche ... 64
Clusters ... 77
Services ... 89
Rôles IAM pour Amazon ECS ... 147
Création de ressources Amazon ECS à l'aide de la AWS CloudFormation console ... 149
Prérequis ... 150
Étape 1 : créer un modèle de pile ... 150
Étape 2 : créer une pile pour les ressources Amazon ECS ... 156
Étape 3 : vérifier ... 157
Étape 4 : Nettoyer les ressources ... 157
Création de ressources Amazon ECS à l'aide de AWS CLI commandes pour AWS CloudFormation ... 158
Prérequis ... 158
Étape 1 : créer une pile ... 158
Étape 2 : vérifier la création des ressources ... 164
Étape 3 : Nettoyer ... 165

Bonnes pratiques 167
AWS Fargate 172
Procédures 172
Fournisseurs de capacité 173
Définitions de tâche 173
Versions de plateforme 173
Équilibrage de charge des services 174
Métriques d'utilisation 174
Considérations relatives à la sécurité concernant le moment d'utiliser le type de lancement Fargate 175
Bonnes pratiques de sécurité à Fargate 175
AWS KMS À utiliser pour chiffrer le stockage éphémère pour Fargate 175
Fonctionnalité SYS_PTRACE pour le suivi des appels système du noyau avec Fargate 176
Utilisez Amazon GuardDuty avec Fargate Runtime Monitoring 176
Considérations relatives à la sécurité de Fargate 177
Versions de la plateforme Fargate 178
..... 179
Migration vers la version 1.4.0 de la plateforme Linux 180
Journal des modifications de version de la plate-forme Linux 181
Obsolète de version de la plate-forme Linux 184
Journal des modifications de version de la plate-forme Windows 186
Considérations relatives aux conteneurs Windows sur Fargate pour Amazon ECS 187
Stockage éphémère de la tâche Fargate 188
Versions de la plateforme de conteneurs Linux Fargate 188
Versions de la plateforme de conteneur Windows Fargate 190
Clés gérées par le client pour un stockage AWS Fargate éphémère 190
Retrait et maintenance des tâches 204
Aperçu de l'avis de retrait des tâches 205
Puis-je me désinscrire de la mise hors service des tâches ? 208
Puis-je recevoir des notifications de retrait de tâches par le biais d'autres AWS services ? .. 209
Puis-je modifier le retrait d'une tâche une fois qu'il a été planifié ? 209
Comment Amazon ECS gère-t-il les tâches faisant partie d'un service ? 209
Amazon ECS peut-il gérer automatiquement des tâches autonomes ? 210
Préparez-vous à la AWS suppression des tâches Fargate sur Amazon ECS 210
AWS Régions de Fargate 213
Conteneurs Linux sur AWS Fargate 213

Conteneurs Windows sur AWS Fargate ... 215
Concevez votre solution pour Amazon ECS ... 218
Capacité ... 219
Points de terminaison de service ... 220
Réseaux ... 220
Accès aux fonctionnalités ... 221
Rôles IAM ... 222
Journalisation ... 223
Types de lancement ... 223
Type de lancement Fargate ... 224
EC2 type de lancement ... 229
Type de lancement externe ... 231
Utilisation des points de terminaison Dual-Stack ... 232
Utilisation des points de terminaison Dual-Stack (AWS CLI) ... 233
Utilisation des points de terminaison Dual-Stack (AWS SDKs) ... 234
Utilisation de points de terminaison Dual-Stack avec l'API REST ... 235
Applications dans des sous-réseaux partagés, des zones locales et des zones de longueur d'onde ... 235
Sous-réseaux partagés ... 236
Zones locales ... 237
Zones Wavelength ... 238
Amazon Elastic Container Service sur AWS Outposts ... 238
Considérations ... 239
Prérequis ... 239
Vue d'ensemble de la création de clusters sur AWS Outposts ... 239
Bonnes pratiques concernant le réseau ... 242
Connectez des applications à Internet ... 243
Meilleures pratiques pour recevoir des connexions entrantes vers Amazon ECS ... 247
Bonnes pratiques en matière de connexion aux AWS services ... 253
Bonnes pratiques pour connecter les services ... 256
Bonnes pratiques pour les services de mise en réseau entre AWS les comptes et VPCs ... 261
AWS services de dépannage réseau ... 262
Optimisation du temps de lancement des tâches ... 263
Fonctionnant à grande échelle ... 265
Quotas de service Amazon ECS et limites de limitation des API ... 265
Gérer les problèmes de régulation ... 271

Meilleures pratiques en matière de mise à l'échelle automatique et de gestion des capacités ... 274
Déterminer la taille de la tâche ... 275
Optimisation de la mise à l'échelle automatique des services ... 276
Capacité et disponibilité ... 283
Capacité du cluster ... 288
Choix de la taille des tâches Fargate ... 289
Accélération de la mise à l'échelle automatique des clusters ... 289
Accédez aux fonctionnalités avec les paramètres du compte ... 291
Amazon Resource Names (ARNs) et IDs ... 292
Calendrier relatif au format ARN et de l'ID de ressource ... 295
Container Insights ... 295
AWS Fargate Conformité à la norme fédérale de traitement de l'information (FIPS-140) ... 297
Autorisation de balisage ... 298
Chronologie des autorisations de balisage ... 299
AWS Fargate temps d'attente pour la retraite des tâches ... 300
Augmenter les interfaces réseau des instances de conteneurs Linux ... 302
Surveillance du temps d'exécution (GuardDuty intégration Amazon) ... 302
IPv6 VPC à double pile ... 303
Mode pilote de journal par défaut ... 304
Affichage de vos paramètres de compte à l'aide de la console ... 304
Modification des paramètres de compte ... 305
Restauration des paramètres par défaut du compte ... 306
Gestion des paramètres du compte à l'aide du AWS CLI ... 307
Rôles IAM pour Amazon ECS ... 309
Définitions de tâches ... 313
États de définition de tâche ... 314
Ressources Amazon ECS pouvant bloquer une suppression ... 315
Architectez votre application ... 316
Bonnes pratiques pour les images de conteneurs ... 318
Bonnes pratiques relatives à la taille des tâches ... 320
Mise en réseau des tâches pour le type de EC2 lancement ... 321
Mise en réseau des tâches pour le type de lancement Fargate ... 334
Options de stockage pour les tâches ... 339
Gestion de l'espace mémoire d'échange de conteneurs ... 430
Différences de définition des tâches pour le type de lancement Fargate ... 432
Différences de définition des tâches pour EC2 les instances exécutant Windows ... 442

Création d'une définition de tâche à l'aide de la console ... 443
Validation JSON ... 443
AWS CloudFormation piles ... 444
Procédure ... 444
Mise à jour d'une définition de tâche à l'aide de la console ... 476
Validation JSON ... 477
Procédure ... 477
Annulation de l'enregistrement d'une révision de définition de tâche à l'aide de la console ... 478
AWS CloudFormation piles ... 444
Procédure ... 479
Suppression de l'enregistrement d'une révision de définition de tâche à l'aide de la console ... 480
Ressources Amazon ECS pouvant bloquer une suppression ... 315
Procédure ... 481
Cas d'utilisation de définition de tâche ... 482
Définitions de tâches pour les charges de travail du GPU ... 482
Définitions de tâches pour les charges de travail de transcodage vidéo ... 493
Définitions de tâches pour les charges de travail d'apprentissage automatique de AWS Neuron ... 507
Définitions de tâches pour les instances de deep learning ... 516
Définitions de tâches pour les charges de travail ARM 64 bits ... 519
Envoyer les journaux à CloudWatch ... 521
Envoyer des journaux à un AWS service ou AWS Partner ... 525
Utilisation d'images n'appartenant pas àAWS un conteneur ... 539
Redémarrer des conteneurs individuels dans les tâches ... 542
Transférer des données sensibles vers un conteneur ... 545
Paramètres de définition des tâches pour le type de lancement Fargate ... 570
Famille ... 570
Types de lancement ... 571
Rôle de tâche ... 571
Rôle d'exécution de tâche ... 571
Mode réseau ... 572
Plateforme d'exécution ... 572
Taille de la tâche ... 573
Définitions de conteneur ... 577
Nom de l'accélérateur Elastic Inference ... 624
Configuration du proxy ... 625

Volumes ... 627
Balises ... 634
Autres paramètres de définition de tâche ... 635
Paramètres de définition des tâches pour le type de EC2 lancement ... 637
Famille ... 638
Types de lancement ... 638
Rôle de tâche ... 639
Rôle d'exécution de tâche ... 639
Mode réseau ... 640
Plateforme d'exécution ... 641
Taille de la tâche ... 642
Définitions de conteneur ... 644
Nom de l'accélérateur Elastic Inference ... 694
Contraintes de placement des tâches ... 695
Configuration du proxy ... 696
Volumes ... 698
Balises ... 705
Autres paramètres de définition de tâche ... 706
Modèle de définition de tâche ... 709
Exemples de définitions de tâches ... 719
serveur Web ... 720
`splunk`pilote de journal ... 722
`fluentd`pilote de journal ... 723
`gelf`pilote de journal ... 724
Charges de travail sur les instances externes ... 724
Image Amazon ECR et définition des tâches (rôle IAM) ... 726
Point d'entrée avec commande ... 726
Dépendances du conteneur ... 727
Volumes dans les définitions de tâches ... 729
Exemples de définitions de tâche Windows ... 730
Clusters ... 731
Fournisseurs de capacité ... 732
Clusters pour Fargate ... 733
Avis de résiliation de Fargate Spot ... 735
Création d'un cluster pour le type de lancement Fargate ... 737
Fournisseurs de capacité pour le type de EC2 lancement ... 740

EC2 sécurité des instances de conteneur ... 742
Création d'un cluster pour le type de EC2 lancement Amazon ... 743
Cluster Auto Scaling ... 749
Instances de EC2 conteneurs Amazon ... 786
Clusters pour le type de lancement externe ... 935
Systèmes d'exploitation et architectures système pris en charge ... 935
Considérations ... 936
Création d'un cluster pour le type de lancement externe ... 940
Enregistrement d'une instance externe dans un cluster Amazon ECS ... 943
Annuler l'enregistrement d'une instance externe ... 949
Mise à jour de l' AWS Systems Manager agent et de l'agent de conteneur Amazon ECS ... 954
Mise à jour d'un cluster ... 959
Suppression d’un cluster ... 961
Annulation de l'enregistrement d'une instance de conteneur ... 962
Procédure ... 963
Drainage des instances de conteneur ... 963
Comportement de drainage pour les services ... 964
Comportement de drainage pour les tâches autonomes ... 965
Procédure ... 965
Instances de EC2 conteneurs Amazon ... 966
Cycle de vie ... 967
Support Docker ... 968
AMI optimisée pour Amazon ECS ... 968
Informations supplémentaires ... 969
Configuration de l'agent de conteneur ... 969
Installation de l'agent de conteneur Amazon ECS ... 973
Paramètres de configuration du journal de l'agent de conteneur ... 979
Configuration des instances de conteneur pour les images Docker privées ... 980
Nettoyer les tâches et les images ... 986
Planifiez vos conteneurs ... 989
Options de calcul ... 991
Cycle de vie d'une tâche ... 992
États de cycle de vie ... 993
Comment Amazon ECS place les tâches sur les instances de conteneur ... 995
EC2 type de lancement ... 995
Type de lancement Fargate ... 996

Utiliser des stratégies pour définir le placement des tâches ... 997
Tâches liées au groupe ... 1003
Définissez les instances de conteneur utilisées pour les tâches ... 1003
Tâches autonomes ... 1015
Flux de travail des tâches ... 1015
Exécution d'une application en tant que tâche ... 1016
Utiliser Amazon EventBridge Scheduler pour planifier des tâches ... 1029
Arrêt d'une tâche ... 1036
Services ... 1037
Option de calcul d'infrastructure ... 1039
Service Auto Scaling ... 1040
Équilibrage de charge des services ... 1040
Services d'interconnexion ... 1042
Contrôleurs et stratégies de déploiement de services ... 1042
Déploiements de services ... 1201
Révisions des services ... 1212
Rééquilibrage des zones de disponibilité ... 1221
Utiliser l'équilibrage de charge pour répartir le trafic de service ... 1231
Service Auto Scaling ... 1246
Services d'interconnexion ... 1289
Protection évolutive des tâches ... 1352
Injection de défauts avec Amazon ECS et Fargate ... 1361
Mettre à jour les paramètres du service Amazon ECS ... 1370
Suppression d'un service ... 1401
Migrer un service (ARN abrégé) ... 1402
Logique de limitation du service ... 1408
Paramètres de définition de service ... 1410
Modèle de définition de service ... 1438
Balisage de ressources ... 1446
Comment les ressources sont étiquetées ... 1447
Balisage des ressources à la création ... 1450
Restrictions ... 1450
Balises gérées par Amazon ECS ... 1451
Utiliser des tags pour la facturation ... 1452
Ajout de balises à des ressources ... 1453
Ajouter des balises à une instance de conteneur ... 1455

Instances de conteneurs externes 1457
Rapports d'utilisation 1458
Utilisation et coût au niveau des tâches 1459
Surveillance 1461
Bonnes pratiques pour la surveillance d'Amazon ECS 1462
Outils de surveillance 1462
Outils automatiques 1462
Outils manuels 1464
Surveillez Amazon ECS à l'aide de CloudWatch 1465
Considérations 1466
Métriques recommandées 1467
Affichage des métriques Amazon ECS 1467
CloudWatch Métriques Amazon ECS 1469
AWS Fargate métriques d'utilisation 1503
Métriques de réservation du cluster Amazon ECS 1505
Mesures d'utilisation du cluster Amazon ECS 1507
Mesures d'utilisation des services Amazon ECS 1509
Automatisez les réponses aux erreurs Amazon ECS à l'aide de EventBridge 1513
Événements Amazon ECS 1514
Gestion des événements 1539
Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée 1543
Déterminer l'état des tâches à l'aide de vérifications de l'état des conteneurs 1544
Comment l'état de santé des tâches est déterminé 1546
Contrôles de santé et déconnexions des agents 1547
Afficher l'état du conteneur 1547
Surveiller l'état de l'instance de conteneur Amazon ECS 1547
Problèmes liés à l'état de santé des instances de conteneurs 1549
Identifiez les opportunités d'optimisation d'Amazon ECS à l'aide des données de suivi des applications 1549
Autorisations IAM requises pour AWS Distro pour OpenTelemetry l'intégration avec AWS X-Ray 1549
Spécification de la AWS distribution pour le OpenTelemetry sidecar à AWS X-Ray intégrer dans la définition de votre tâche 1551
Corrélez les performances des applications Amazon ECS à l'aide des métriques des applications 1553

Exporter les métriques des applications vers Amazon CloudWatch 1553
Exportation des métriques d'application vers Amazon Managed Service for Prometheus ... 1558
Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail 1562
Événements de gestion Amazon ECS dans CloudTrail 1564
Exemples d'événements Amazon ECS 1564
Affichage CloudWatch des journaux en direct pour les services et tâches Amazon ECS 1566
Autorisations requises 1567
Procédure 1567
AWS CLI références 1568
Surveillez les charges de travail à l'aide de métadonnées 1568
Variables d’environnement 1569
Fichier de métadonnées de conteneur 1570
Métadonnées de tâches disponibles pour les tâches Amazon ECS sur EC2 1577
Métadonnées de tâches disponibles pour les tâches sur Fargate 1621
Introspection des conteneurs 1645
Identifiez les comportements non autorisés grâce à la surveillance du temps d'exécution 1648
Comment fonctionne la surveillance du temps d'exécution avec Amazon ECS 1649
Considérations 1650
Utilisation des ressources 1650
Surveillance du temps d'exécution pour les charges de travail Fargate 1651
Surveillance du temps d'exécution pour les charges EC2 de travail 1655
Résolution des problèmes de surveillance du temps 1660
Surveillez les conteneurs Amazon ECS avec ECS Exec 1664
Considérations 1664
Prérequis 1667
Architecture 1667
Utilisation d'ECS Exec 1668
Journalisation à l'aide d'ECS Exec 1671
Utilisation de stratégies IAM pour limiter l'accès à ECS Exec 1675
Recommandations Compute Optimizer 1679
Recommandations relatives à la taille des tâches pour Fargate 1679
Résolution des problèmes 1680
Résoudre les erreurs liées aux tâches arrêtées 1683
Mises à jour des messages d'erreur des tâches arrêtées 1684
Affichage des erreurs liées aux tâches arrêtées 1689
Messages d'erreur relatifs aux tâches arrêtées 1691

Vérification de la connectivité des tâches ... 1712
Afficher les demandes de rôle IAM ... 1718
Affichage des messages relatifs aux événements de service ... 1719
Messages relatifs aux événements du service Amazon ECS ... 1720
Messages d'événements non conformes au service Amazon ECS ... 1732
Messages relatifs aux événements du service de rééquilibrage du service Amazon ECS Availability Zone ... 1733
Résolution des problèmes liés aux équilibreurs de charge de service dans Amazon ECS ... 1735
Résolution des problèmes liés au dimensionnement automatique du service dans Amazon ECS ... 1737
Résoudre les erreurs de processeur ou de mémoire non valides liées à la définition des tâches ... 1738
Afficher les journaux des agents de conteneur ... 1740
Collecte des journaux de conteneurs avec le collecteur de journaux Amazon ECS ... 1742
Introspection de l'agent ... 1745
Diagnostic Docker dans Amazon ECS ... 1747
Répertorier les conteneurs Docker dans Amazon ECS ... 1747
Afficher les journaux Docker dans Amazon ECS ... 1748
Inspectez les conteneurs Docker dans Amazon ECS ... 1749
Configuration de la sortie détaillée du démon Docker dans Amazon ECS ... 1750
Résoudre les problèmes liés au Docker dans `API error (500): devmapper` Amazon ECS ... 1752
Résoudre les problèmes liés à ECS Exec ... 1753
Vérifiez à l'aide de l'Exec Checker ... 1754
Erreur lors de l'appel à `execute-command` ... 1754
Résoudre les problèmes liés à Amazon ECS Anywhere ... 1754
Problèmes d'enregistrement d'instance externe ... 1755
Problèmes de réseau d'instance externe ... 1756
Problèmes d'exécution de tâches ... 1756
AWS Fargate limitation des quotas ... 1756
Limiter l'`RunTask`API dans Fargate ... 1757
Ajustement des quotas tarifaires dans Fargate ... 1758
Motifs d'échec d'API ... 1758
Sécurité ... 1770
Gestion de l’identité et des accès ... 1771
Public ciblé ... 1771

Authentification par des identités ... 1772
Gestion des accès à l'aide de politiques ... 1776
Fonctionnement d'Amazon Elastic Container Service avec IAM ... 1779
Exemples de politiques basées sur l'identité ... 1792
AWS politiques gérées pour Amazon ECS ... 1804
Utilisation des rôles liés à un service ... 1819
Rôles IAM pour Amazon ECS ... 1823
Autorisations requises pour la console Amazon ECS ... 1890
Autorisations requises pour les fonctions Lambda dans les déploiements Amazon ECS blue/green ... 1900
Autorisations IAM requises pour le dimensionnement automatique du service Amazon ECS ... 1901
Baliser des ressources pendant la création ... 1902
Résolution des problèmes ... 1907
Bonnes pratiques IAM ... 1909
Journalisation et surveillance ... 1912
Validation de conformité ... 1914
Bonnes pratiques en matière de conformité et de sécurité ... 1916
AWS Fargate Conformité à la norme FIPS-140 ... 1918
AWS Fargate Considérations relatives à la norme FIPS-140 ... 1918
Utilisation de FIPS sur Fargate ... 1919
Utilisation CloudTrail pour l'audit Fargate FIPS-140 ... 1919
Sécurité de l'infrastructure ... 1921
Points de terminaison d'un VPC d'interface (AWS PrivateLink) ... 1922
Modèle de responsabilité partagée ... 1929
Type de lancement Fargate ... 1929
EC2 type de lancement ... 1930
Bonnes pratiques en matière de sécurité réseau ... 1931
Chiffrement en transit ... 1931
Mise en réseau des tâches ... 1933
AWS PrivateLink et Amazon ECS ... 1933
Paramètres de l'agent de conteneur ... 1935
Recommandations relatives à la sécurité du réseau ... 1935
Bonnes pratiques en matière de sécurité des tâches et des conteneurs ... 1937
Créez un minimum d'images ou utilisez des images sans distribution ... 1937
Analysez vos images pour détecter les vulnérabilités ... 1938

Supprimez les autorisations spéciales associées à vos images 1940
Créez un ensemble d'images sélectionnées 1940
Analysez les packages d'applications et les bibliothèques pour détecter les vulnérabilités . 1938
Effectuez une analyse de code statique 1941
Exécutez les conteneurs en tant qu'utilisateur non root 1941
Utilisez un système de fichiers racine en lecture seule 1942
Configuration des tâches avec des limites de processeur et de mémoire (Amazon EC2) ... 1942
Utilisez des balises immuables avec Amazon ECR 1943
Évitez de gérer les conteneurs en tant que privilégiés (Amazon EC2) 1943
Supprimez les fonctionnalités Linux inutiles du conteneur 1943
Utilisez une clé gérée par le client (CMK) pour chiffrer les images transférées vers Amazon ECR 1944
Didacticiels 1945
Création d'une tâche Linux pour le type de lancement Fargate avec AWS CLI 1948
Prérequis 1948
Étape 1 : Créer un cluster 1949
Étape 2 : Enregistrer une définition de tâche Linux 1950
Étape 3 : Répertorier les définitions de tâche 1952
Étape 4 : Créer un service 1952
Étape 5 : Répertorier les services 1953
Étape 6 : Décrire le service en cours d'exécution 1953
Étape 7 : Test 1956
Étape 8 : Nettoyer 1959
Création d'une tâche Windows pour le type de lancement Fargate avec AWS CLI 1960
Prérequis 1960
Étape 1 : Créer un cluster 1961
Étape 2 : Enregistrement d'une définition de tâche Windows 1962
Étape 3 : Répertorier les définitions de tâche 1963
Étape 4 : Créer un service 1964
Étape 5 : Répertorier les services 1964
Étape 6 : Décrire le service en cours d'exécution 1965
Étape 7 : Nettoyer 1967
Création d'une tâche pour le type de EC2 lancement avec AWS CLI 1967
Prérequis 1968
Créer un cluster 1969
Lancer une instance de conteneur avec l'AMI Amazon ECS 1969

Lister les instances de conteneur 1972
Décrivez votre instance de conteneur 1972
Enregistrer une définition de tâche 1975
Lister les définitions de tâches 1977
Créer un service 1978
Lister les services 1979
Décrivez le service 1979
Décrire la tâche en cours 1980
Testez le serveur Web 1981
Nettoyage des ressources 1982
Configuration d'Amazon ECS pour écouter CloudWatch les événements 1984
Prérequis : Configuration d'un cluster test 1984
Étape 1 : Créer la fonction Lambda 1984
Étape 2 : Enregistrer une règle d'événement 1985
Étape 3 : Créer une définition de tâche 1986
Étape 4 : Test de la règle 1987
Envoi d'alertes Amazon Simple Notification Service en cas d'arrêt d'une tâche 1988
Prérequis : Configuration d'un cluster test 1988
Prérequis : configurer les autorisations pour Amazon SNS 1988
Étape 1 : Créer une rubrique Amazon SNS et s'y abonner 1989
Étape 2 : Enregistrer une règle d'événement 1989
Étape 3 : Test de la règle 1990
Concaténation de messages de journal multilignes ou de stack-trace 1992
Autorisations IAM requises 1992
Déterminez à quel moment utiliser le paramètre de journal multiligne 1994
Options d'analyse et de concaténation 1995
Déploiement de Fluent Bit sur des conteneurs Windows 2015
Prérequis 2018
Étape 1 : Créer les rôles IAM d'accès 2018
Étape 2 : Créer une instance de conteneur Windows Amazon ECS 2019
Étape 3 : Configurer Fluent Bit 2020
Étape 4 : enregistrer une définition de tâche Windows Fluent Bit qui achemine les journaux vers CloudWatch 2022
Étape 5 : Exécuter la définition de tâche `ecs-windows-fluent-bit` en tant que service Amazon ECS en utilisant la stratégie de planification du démon 2024
Étape 6 : Enregistrer une définition de tâche Windows qui génère les journaux 2025

Étape 7 : Créer la définition de tâche `windows-app-task` 2027
Étape 8 : Vérifiez les connexions CloudWatch 2028
Étape 9 : nettoyer 2029
Utilisation gMSA pour les EC2 Linux conteneurs 2030
Considérations 2030
Prérequis 2031
Configuration 2033
CredSpec file 2040
Utilisation gMSA pour les Linux conteneurs sur Fargate 2041
Considérations 2041
Prérequis 2042
Configuration 2043
CredSpec file 2046
Utilisation de conteneurs Windows et de conteneurs sans domaine à gMSA l'aide du AWS CLI 2047
Prérequis 2048
Étape 1 : créer et configurer le compte gMSA sur les services de domaine Active Directory (AD DS) 2050
Étape 2 : charger les informations d'identification dans Secrets Manager 2052
Étape 3 : modifier votre JSON CredSpec pour inclure des informations gMSA sans domaine 2053
Étape 4 : charger CredSpec dans Amazon S3 2054
Étape 5 (facultative) : créer un cluster Amazon ECS 2055
Étape 6 : créer un rôle IAM pour les instances de conteneur 2056
Étape 7 : créer un rôle d'exécution de tâche personnalisé 2056
Étape 8 : créer un rôle de tâche pour Amazon ECS Exec 2057
Étape 9 : enregistrer une définition de tâche 2059
Étape 10 : enregistrer une instance de conteneur Windows 2061
Étape 11 : vérifier l'instance de conteneur 2061
Étape 12 : exécuter une tâche Windows 2063
Étape 13 : vérifier que le conteneur possède les informations d'identification gMSA 2063
Étape 14 : nettoyer 2064
Débogage 2066
Apprenez à utiliser g MSAs pour les conteneurs EC2 Windows 2066
Considérations 2067
Prérequis 2068

Configuration 2069
Utilisation d'Image Builder pour créer une solution personnalisée optimisée pour Amazon ECS AMIs 2075
Utilisation de l'ARN de l'image avec l'infrastructure en tant que code (IaC) 2077
Utilisation de l'ARN de l'image avec AWS CloudFormation 2079
Utilisation de l'ARN de l'image avec Terraform 2081
Utilisation de AWS Deep Learning Containers 2081
Service Quotas 2082
Gérer vos quotas de service dans le AWS Management Console 2082
Gérez les quotas de service et les limites de limitation des API 2084
Elastic Load Balancing 267
Interfaces réseau Elastic 268
AWS Cloud Map 270
Référence d'API Amazon ECS 2090
Historique du document 2091
..... mmcxliv

Qu'est-ce qu'Amazon Elastic Container Service ?

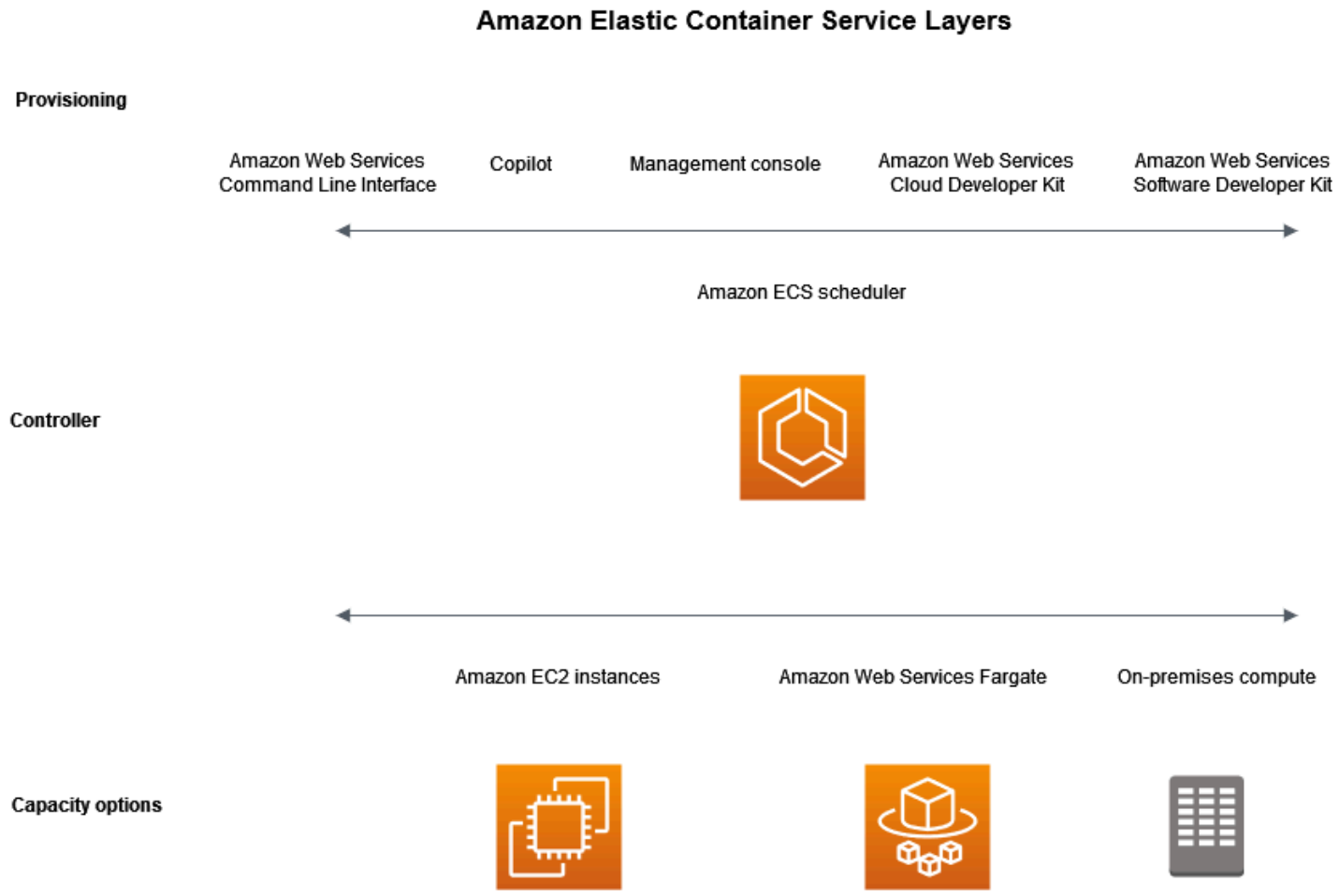
Amazon Elastic Container Service (Amazon ECS) est un service d'orchestration de conteneurs entièrement géré qui vous permet de déployer, de gérer et de dimensionner aisément des applications conteneurisées. En tant que service entièrement géré, Amazon ECS intègre les meilleures pratiques opérationnelles et de AWS configuration. Il est intégré à la fois à AWS des outils, tels qu'Amazon Elastic Container Registry, et à des outils tiers, tels que Docker. Cette intégration permet aux équipes de se concentrer plus facilement sur la création des applications, et non sur l'environnement. Vous pouvez exécuter et faire évoluer vos charges de travail de conteneurs Régions AWS dans le cloud et sur site, sans la complexité liée à la gestion d'un plan de contrôle.

Terminologie et composants

Amazon ECS comporte trois couches :

- Capacité : l'infrastructure dans laquelle vos conteneurs sont exécutés
- Contrôleur : déploiement et gestion de vos applications qui s'exécutent sur les conteneurs
- Provisionnement : les outils que vous pouvez utiliser pour interagir avec le planificateur afin de déployer et de gérer vos applications et vos conteneurs

Le schéma suivant illustre les couches Amazon ECS.



La capacité est l'infrastructure dans laquelle vos conteneurs sont exécutés. Voici un aperçu des options de capacité :

- EC2 Instances Amazon dans le AWS cloud

  Vous choisissez le type d'instance, le nombre d'instances et vous gérez la capacité.
- Sans serveur (AWS Fargate) dans le cloud AWS

  Fargate est un moteur de calcul sans serveur pay-as-you-go. Avec Fargate, vous n'avez pas besoin de gérer les serveurs, de gérer la planification de la capacité ou d'isoler les charges de travail des conteneurs pour des raisons de sécurité.
- Machines virtuelles (VM) ou serveurs sur site

  Amazon ECS Anywhere fournit un support pour l'enregistrement d'une Instance externe, telle qu'un serveur sur site ou une machine virtuelle (VM), sur votre cluster Amazon ECS.

Le planificateur Amazon ECS est le logiciel qui gère vos applications.

Fonctionnalités

Amazon ECS fournit les fonctionnalités de haut niveau suivantes :

Définition de tâche

Le plan de l'application.

Cluster

L'infrastructure sur laquelle votre application s'exécute.

Tâche

Application telle qu'un traitement par lots qui exécute un travail, puis s'arrête.

Service

Une application apatride de longue durée.

Paramétrage du compte

Permet d'accéder aux fonctionnalités.

Cluster Auto Scaling

Amazon ECS gère le dimensionnement des EC2 instances Amazon enregistrées dans votre cluster.

Service Auto Scaling

Amazon ECS augmente ou diminue automatiquement le nombre de tâches souhaité dans votre service.

Allouer

Il existe plusieurs options pour le provisionnement d'Amazon ECS :

- AWS Management Console — Offre une interface web que vous pouvez utiliser pour accéder à vos ressources Amazon ECS.
- AWS Command Line Interface (AWS CLI) — Fournit des commandes pour un large éventail de AWS services, y compris Amazon ECS. Elle est prise en charge sur Windows, Mac et Linux. Pour de plus amples informations, veuillez consulter AWS Command Line Interface.

- AWS SDKs— Fournit des informations spécifiques à la langue APIs et prend en charge de nombreux détails de connexion. Ces outils incluent le calcul des signatures, la gestion des nouvelles tentatives de demande et la gestion des erreurs. Pour de plus amples informations, veuillez consulter AWS SDKs.
- AWS CDK— Fournit un cadre de développement logiciel open source que vous pouvez utiliser pour modéliser et allouer vos ressources d'applications cloud à l'aide de langages de programmation familiers. Le AWS CDK alloue vos ressources de manière sûre et répétable grâce à AWS CloudFormation.

Tarification

La tarification d'Amazon ECS dépend de l'option de capacité que vous choisissez pour vos conteneurs.

- Tarification Amazon ECS : informations sur la tarification pour Amazon ECS.
- AWS Fargate tarification — Informations sur les prix de Fargate.

Services connexes

Services à utiliser avec Amazon ECS

Vous pouvez utiliser other Services AWS pour vous aider à déployer vos tâches et services sur Amazon ECS.

Amazon EC2 Auto Scaling

Permet de garantir que vous disposez du nombre correct d' EC2 instances Amazon disponibles pour gérer la charge de votre application.

Amazon CloudWatch

Surveillez vos services et vos tâches.

Amazon Elastic Container Registry

Stockez et gérez les images des conteneurs.

Elastic Load Balancing

Répartissez automatiquement le trafic de service entrant.

Amazon GuardDuty

Détectez les utilisations potentiellement non autorisées ou malveillantes de vos instances de conteneur et de vos charges de travail.

Découvrez comment créer et utiliser les ressources Amazon ECS

Les guides suivants présentent les outils disponibles pour accéder à Amazon ECS ainsi que les procédures d'introduction pour exécuter des conteneurs. Docker Basics vous permet d'effectuer les étapes de base pour créer une image de conteneur Docker et la charger dans un référentiel privé Amazon ECR. Les guides de démarrage vous expliquent comment utiliser l'interface de ligne de commande AWS Copilot et comment effectuer les AWS Management Console tâches courantes relatives à l'exécution de vos conteneurs sur Amazon ECS et. AWS Fargate

Table des matières

- Configurer l'utilisation d'Amazon ECS
- Création d'une image de conteneur à utiliser sur Amazon ECS
- Découvrez comment créer une tâche Linux Amazon ECS pour le type de lancement Fargate
- Découvrez comment créer une tâche Windows Amazon ECS pour le type de lancement Fargate
- Découvrez comment créer une tâche Windows Amazon ECS pour le type de EC2 lancement
- Création de ressources Amazon ECS à l'aide du AWS CDK
- Création de ressources Amazon ECS à l'aide de l'interface de ligne de commande AWS Copilot

Configurer l'utilisation d'Amazon ECS

Si vous êtes déjà inscrit à Amazon Web Services (AWS) et que vous utilisez Amazon Elastic Compute Cloud (Amazon EC2), vous êtes sur le point de pouvoir utiliser Amazon ECS. Le processus de configuration est très similaire pour les deux services. Le guide suivant vous prépare au lancement de votre premier cluster Amazon ECS.

Effectuez les tâches suivantes pour vous préparer à utiliser Amazon ECS.

AWS Management Console

AWS Management Console Il s'agit d'une interface basée sur un navigateur permettant de gérer les ressources Amazon ECS. Elle offre une vue d'ensemble du service, ce qui facilite l'exploration des fonctions et rôles d'Amazon ECS sans avoir besoin d'outils supplémentaires. Une multitude de didacticiels et de démonstrations sont disponibles pour vous guider dans l'utilisation de la console.

Pour obtenir un didacticiel pour vous guider dans la console, consultez Découvrez comment créer et utiliser les ressources Amazon ECS.

Au début, de nombreux clients préfèrent utiliser la console, car elle fournit des informations visuelles instantanées sur le succès des actions entreprises. AWS les clients qui les connaissent peuvent facilement gérer AWS Management Console les ressources associées telles que les équilibreurs de charge et les EC2 instances Amazon.

Commencez par le AWS Management Console.

Inscrivez-vous pour un Compte AWS

Si vous n'en avez pas Compte AWS, procédez comme suit pour en créer un.

Pour vous inscrire à un Compte AWS

1. Ouvrez l'https://portal.aws.amazon.com/billing/inscription.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d'inscription, vous recevrez un appel téléphonique ou un SMS et vous saisirez un code de vérification en utilisant le clavier numérique du téléphone.

   Lorsque vous vous inscrivez à un Compte AWS, un Utilisateur racine d'un compte AWSest créé. Par défaut, seul l'utilisateur racine a accès à l'ensemble des Services AWS et des ressources de ce compte. La meilleure pratique de sécurité consiste à attribuer un accès administratif à un utilisateur, et à utiliser uniquement l'utilisateur racine pour effectuer les tâches nécessitant un accès utilisateur racine.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. À tout moment, vous pouvez consulter l'activité actuelle de votre compte et gérer votre compte en accédant à https://aws.amazon.com/et en choisissant Mon compte.

Création d'un utilisateur doté d'un accès administratif

Une fois que vous vous êtes inscrit à un utilisateur administratif Compte AWS, que vous Utilisateur racine d'un compte AWS l'avez sécurisé AWS IAM Identity Center, que vous l'avez activé et que vous en avez créé un, afin de ne pas utiliser l'utilisateur root pour les tâches quotidiennes.

Sécurisez votre Utilisateur racine d'un compte AWS

1. Connectez-vous en AWS Management Consoletant que propriétaire du compte en choisissant Utilisateur root et en saisissant votre adresse Compte AWS e-mail. Sur la page suivante, saisissez votre mot de passe.

   Pour obtenir de l'aide pour vous connecter en utilisant l'utilisateur racine, consultez Connexion en tant qu'utilisateur racine dans le Guide de l'utilisateur Connexion à AWS .
2. Activez l'authentification multifactorielle (MFA) pour votre utilisateur racine.

   Pour obtenir des instructions, consultez la section Activer un périphérique MFA virtuel pour votre utilisateur Compte AWS root (console) dans le guide de l'utilisateur IAM.

Création d'un utilisateur doté d'un accès administratif

1. Activez IAM Identity Center.

   Pour obtenir des instructions, consultez Activation d' AWS IAM Identity Center dans le Guide de l'utilisateur AWS IAM Identity Center .
2. Dans IAM Identity Center, octroyez un accès administratif à un utilisateur.

   Pour un didacticiel sur l'utilisation du Répertoire IAM Identity Center comme source d'identité, voir Configurer l'accès utilisateur par défaut Répertoire IAM Identity Center dans le Guide de AWS IAM Identity Center l'utilisateur.

Connexion en tant qu'utilisateur doté d'un accès administratif

- Pour vous connecter avec votre utilisateur IAM Identity Center, utilisez l'URL de connexion qui a été envoyée à votre adresse e-mail lorsque vous avez créé l'utilisateur IAM Identity Center.

  Pour obtenir de l'aide pour vous connecter en utilisant un utilisateur d'IAM Identity Center, consultez la section Connexion au portail AWS d'accès dans le guide de l'Connexion à AWS utilisateur.

Attribution d'un accès à d'autres utilisateurs

1. Dans IAM Identity Center, créez un ensemble d'autorisations qui respecte la bonne pratique consistant à appliquer les autorisations de moindre privilège.

Pour obtenir des instructions, consultez Création d'un ensemble d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .

2. Attribuez des utilisateurs à un groupe, puis attribuez un accès par authentification unique au groupe.

   Pour obtenir des instructions, consultez Ajout de groupes dans le Guide de l'utilisateur AWS IAM Identity Center .

Créer un Virtual Private Cloud

Vous pouvez utiliser Amazon Virtual Private Cloud (Amazon VPC) pour lancer AWS des ressources dans un réseau virtuel que vous avez défini. Nous vous recommandons de lancer vos instances de conteneurs dans un VPC.

Si vous avez un VPC par défaut, vous pouvez ignorer cette section et passer à la tâche suivante, Création d'un groupe de sécurité. Pour déterminer si vous disposez d'un VPC par défaut, consultez la section Utilisation de votre VPC par défaut et de vos sous-réseaux par défaut dans le guide de l'utilisateur Amazon VPC. Sinon, vous pouvez créer dans votre compte un VPC autre qu'un VPC par défaut en suivant les étapes ci-après.

Pour plus d'informations sur la création d'un VPC, consultez la section Créer un VPC dans le guide de l'utilisateur Amazon VPC, et utilisez le tableau suivant pour déterminer les options à sélectionner.

Option	Valeur
Ressources à créer	VPC uniquement
Nom	Vous pouvez, si vous le souhaitez, nommer votre VPC.
IPv4 Bloc d'adresse CIDR	IPv4 Entrée manuelle CIDR La taille du bloc d'adresse CIDR doit être comprise entre /16 et /28.
IPv6 Bloc d'adresse CIDR	Aucun IPv6 bloc CIDR

Option	Valeur	
Location	Par défaut	

Pour plus d'informations sur Amazon VPC, consultez Qu'est-ce qu'Amazon VPC ? dans le Guide de l'utilisateur Amazon VPC.

Création d'un groupe de sécurité

Les groupes de sécurité font office de pare-feu pour les instances de conteneur associées, en contrôlant le trafic entrant et le trafic sortant au niveau de l'instance de conteneur. Vous pouvez ajouter des règles à un groupe de sécurité qui vous permettent de vous connecter à votre instance de conteneur à partir de votre adresse IP avec SSH. Vous pouvez aussi ajouter des règles qui permettent les accès HTTP et HTTPS entrants et sortants depuis n'importe quel emplacement. Ajoutez des règles pour ouvrir les ports qui sont requis par vos tâches. Les instances de conteneur ont besoin d'un accès réseau externe pour communiquer avec le point de terminaison d'Amazon ECS service.

Si vous prévoyez de lancer des instances de conteneur dans plusieurs régions, vous devez créer un groupe de sécurité dans chaque région. Pour plus d'informations, consultez la section Régions et zones de disponibilité dans le guide de EC2 l'utilisateur Amazon.

> **Tip**
>
> Vous avez besoin de l'adresse IP publique de votre ordinateur local, que vous pouvez obtenir à l'aide d'un service. Par exemple, nous fournissons le service suivant : http://checkip.amazonaws.com/ ou https://checkip.amazonaws.com/. Pour trouver un autre service qui fournit votre adresse IP, utilisez l'expression de recherche « what is my IP address » (quelle est mon adresse IP). Si votre connexion s'effectue via un fournisseur d'accès Internet (FAI) ou derrière un pare-feu sans adresse IP statique, vous devez déterminer la plage d'adresses IP utilisée par les ordinateurs clients.

Pour plus d'informations sur la création d'un groupe de sécurité, consultez la section Créer un groupe de sécurité pour votre EC2 instance Amazon dans le guide de EC2 l'utilisateur Amazon et utilisez le tableau suivant pour déterminer les options à sélectionner.

Option	Valeur
Région	La même région que celle que vous avez créée avec votre paire de clés.
Nom	Un nom facile à retenir, tel que ecs-instances-default-cluster.
VPC	Le VPC par défaut (marqué par « (default) »). **Note** Si votre compte prend en charge Amazon EC2 Classic, sélection nez le VPC que vous avez créé lors de la tâche précédente.

Pour plus d'informations sur les règles de trafic sortant à ajouter pour vos cas d'utilisation, consultez la section Règles des groupes de sécurité pour les différents cas d'utilisation dans le guide de l' EC2 utilisateur Amazon.

Les instances de conteneur Amazon ECS ne nécessitent pas que tous les ports entrants soient ouverts. Vous pouvez toutefois ajouter une règle SSH, afin de vous connecter à l'instance de conteneur et d'examiner les tâches avec les commandes Docker. Vous pouvez également ajouter des règles pour HTTP et HTTPS si vous voulez que votre instance de conteneur héberge une tâche qui exécute un serveur web. Les instances de conteneur ont besoin d'un accès réseau externe pour communiquer avec le point de terminaison d'Amazon ECS service. Appliquez la procédure suivante pour ajouter ces règles de groupe de sécurité facultatives.

Ajoutez les trois règles entrantes suivantes à votre groupe de sécurité. Pour plus d'informations sur la création d'un groupe de sécurité, consultez Configurer les règles du groupe de sécurité dans le guide de l'utilisateur Amazon EC2 .

Option	Valeur	
Règle HTTP	Type : HTTP Source : n'importe où (`0.0.0.0/0`) Cette option ajoute automatiquement le bloc IPv4 CIDR 0.0.0.0/0 comme source. Cette solution est acceptable pour une brève durée dans un environnement de test, mais n'est pas sûre dans des environnements de production. Dans un environnement de production, vous autorisez uniquement l'accès à votre instance pour une adresse IP ou une plage d'adresses spécifiques.	
Règle HTTP	Type : HTTPS Source : n'importe où (`0.0.0.0/0`) Cette solution est acceptable pour une brève durée dans un environnement de test, mais n'est pas sûre dans des environnements de production. Dans un environnement de production, vous autorisez uniquement l'accès à votre instance pour une adresse	

Option	Valeur
	IP ou une plage d'adresses spécifiques.
Règle SSH	Type : SSH Source : choisissez Custom (Personnalisée) et spécifiez l'adresse IP publique de votre ordinateur ou réseau en notation CIDR. Pour spécifier une adresse IP individuelle en notation CIDR, ajoutez le préfixe de routage `/32`. Par exemple, si votre adresse IP est `203.0.113.25` , spécifiez `203.0.113.25/32` . Si votre entreprise alloue des adresses à partir d'une plage, spécifiez la plage complète, telle que `203.0.113.0/24` . **Important** Pour des raisons de sécurité, il est déconseillé d'autoriser l'accès de SSH à votre instance à partir de toutes les adresses IP (`0.0.0.0/0`), si ce n'est à titre de test et pour une très brève durée.

Créez les informations d'identification pour vous connecter à votre EC2 instance

Pour Amazon ECS, une paire de clés n'est nécessaire que si vous avez l'intention d'utiliser le type de EC2 lancement.

AWS utilise le chiffrement à clé publique pour sécuriser les informations de connexion de votre instance. Une instance Linux, comme une instance de conteneur Amazon ECS, n'a aucun mot de passe à utiliser pour l'accès SSH. Une paire de clés vous permet de vous connecter en toute sécurité à votre instance. Vous devez indiquer le nom de la paire de clés au lancement de votre instance de conteneur, puis fournir la clé privée lorsque vous vous connectez avec SSH.

Si vous n'avez pas encore créé de paire de clés, vous pouvez en créer une à l'aide de la EC2 console Amazon. Si vous prévoyez de lancer des instances dans plusieurs régions, vous devez créer une paire de clés dans chaque région. Pour plus d'informations sur les régions, consultez Régions et zones de disponibilité dans le guide de EC2 l'utilisateur Amazon.

Création d’une paire de clés

- Utilisez la EC2 console Amazon pour créer une paire de clés. Pour plus d'informations sur la création d'une paire de clés, consultez la section Créer une paire de clés dans le guide de EC2 l'utilisateur Amazon.

Pour plus d'informations sur la connexion à votre instance, consultez Connect to your Linux instance dans le guide de EC2 l'utilisateur Amazon.

Installez le AWS CLI

Il AWS Management Console peut être utilisé pour gérer toutes les opérations manuellement avec Amazon ECS. Cependant, vous pouvez l'installer AWS CLI sur votre poste de travail local ou sur un boîtier de développement afin de créer des scripts capables d'automatiser les tâches de gestion courantes dans Amazon ECS.

Pour l'utiliser AWS CLI avec Amazon ECS, installez la dernière AWS CLI version. Pour plus d'informations sur l'installation AWS CLI ou la mise à niveau vers la dernière version, voir Installation ou mise à jour vers la dernière version du AWS CLI Guide de l'AWS Command Line Interface utilisateur.

Le AWS Command Line Interface (AWS CLI) est un outil unifié que vous pouvez utiliser pour gérer vos AWS services. Avec cet outil unique, vous pouvez à la fois contrôler plusieurs AWS services et automatiser ces services par le biais de scripts. Les commandes Amazon ECS figurant dans le AWS CLI sont le reflet de l'API Amazon ECS.

AWS CLI Il convient aux clients qui préfèrent et sont habitués à créer des scripts et à s'interfacer avec un outil de ligne de commande et qui savent exactement quelles actions ils souhaitent effectuer sur leurs ressources Amazon ECS. AWS CLI Il est également utile aux clients qui souhaitent se familiariser avec Amazon ECS APIs. Les clients peuvent utiliser le AWS CLI pour effectuer un certain nombre d'opérations sur les ressources Amazon ECS, notamment les opérations de création, de lecture, de mise à jour et de suppression, directement depuis l'interface de ligne de commande.

Utilisez le AWS CLI si vous connaissez ou souhaitez vous familiariser avec Amazon ECS APIs et les commandes CLI correspondantes et si vous souhaitez écrire des scripts automatisés et effectuer des actions spécifiques sur les ressources Amazon ECS.

AWS fournit également les outils de ligne de commande AWS Tools for Windows PowerShell. Pour plus d’informations, consultez le Guide de l’utilisateur AWS Tools for Windows PowerShell.

Étapes suivantes pour utiliser Amazon ECS

Après l'avoir installé AWS CLI, il existe de nombreux outils que vous pouvez utiliser tout en continuant à utiliser Amazon ECS. Les liens suivants expliquent ce que sont certains de ces outils et donnent des exemples de leur utilisation avec Amazon ECS.

- Créez votre première image de conteneur avec Docker et envoyez-la vers Amazon ECR pour l'utiliser dans vos définitions de tâches Amazon ECS.
- Découvrez comment créer une tâche Linux Amazon ECS pour le type de lancement Fargate.
- Découvrez comment créer une tâche Windows Amazon ECS pour le type de lancement Fargate.
- Découvrez comment créer une tâche Windows Amazon ECS pour le type de EC2 lancement.
- À l'aide de votre langage de programmation préféré, définissez l'infrastructure ou l'architecture en tant que code avec Création de ressources Amazon ECS à l'aide du AWS CDK.
- Définissez et gérez toutes les AWS ressources de votre environnement grâce à un déploiement automatisé à l'aide deUtilisation d'Amazon ECS avec AWS CloudFormation.
- Utilisez le flux de travail complet des Création de ressources Amazon ECS à l'aide de l'interface de ligne de commande AWS Copilot end-to-end développeurs pour créer, publier et exploiter des applications de conteneur conformes aux AWS meilleures pratiques en matière d'infrastructure.

Création d'une image de conteneur à utiliser sur Amazon ECS

Amazon ECS utilise des images Docker dans les définitions de tâches pour lancer des conteneurs. Docker est une technologie qui fournit les outils nécessaires pour créer, exécuter, tester et déployer des applications distribuées sur des conteneurs.

Amazon ECS planifie les applications conteneurisées sur des instances de conteneur ou sur AWS Fargate. Les applications conteneurisées sont conditionnées sous forme d'images de conteneur. Cet exemple crée une image de conteneur pour un serveur Web.

Vous pouvez créer votre première image Docker, puis transférer cette image vers Amazon ECR, qui est un registre de conteneurs, pour l'utiliser dans vos définitions de tâches Amazon ECS. Ce guide part du principe que vous possédez des connaissances élémentaires de Docker et de son fonctionnement. Pour plus d'informations sur Docker, consultez Qu'est-ce que Docker ? et la documentation Docker.

Prérequis

Avant de commencer, assurez-vous de respecter les prérequis suivants :

- Veillez à avoir terminé les étapes de configuration d'Amazon ECR. Pour plus d'informations, consultez la section Déplacement d'une image tout au long de son cycle de vie dans Amazon ECR dans le guide de l'utilisateur d'Amazon Elastic Container Registry.
- Votre utilisateur dispose des autorisations IAM requises pour accéder au service Amazon ECR et l'utiliser. Pour en savoir plus, consultez Politiques gérées Amazon ECR.
- Vous avez installé Docker. Pour connaître les étapes d'installation de Docker pour Amazon Linux 2023, consultezInstallation de Docker sur 023 AL2. Pour tous les autres systèmes d'exploitation, consultez la documentation Docker dans Présentation de Docker Desktop.
- Vous l'avez AWS CLI installé et configuré. Pour plus d'informations, voir Installation ou mise à jour vers la dernière version du AWS CLI dans le guide de AWS Command Line Interface l'utilisateur.

Si vous n'avez pas ou n'avez pas besoin d'un environnement de développement local et que vous préférez utiliser une EC2 instance Amazon pour utiliser Docker, nous vous proposons les étapes suivantes pour lancer une EC2 instance Amazon avec Amazon Linux 2023 et installer Docker Engine et la CLI Docker.

Installation de Docker sur 023 AL2

Docker est disponible sur plusieurs systèmes d'exploitation, notamment les distributions Linux les plus modernes, comme Ubuntu et même MacOS et Windows. Pour en savoir plus sur la façon d'installer Docker sur votre système d'exploitation, consultez le guide d'installation Docker.

Vous n'avez pas besoin d'un système de développement local pour utiliser Docker. Si vous utilisez EC2 déjà Amazon, vous pouvez lancer une instance Amazon Linux 2023 et installer Docker pour commencer.

Si vous avez déjà installé Docker, passez à Créer une image Docker.

Pour installer Docker sur une EC2 instance Amazon à l'aide d'une AMI Amazon Linux 2023

1. Lancez une instance avec la dernière AMI Amazon Linux 2023. Pour plus d'informations, consultez Lancer une EC2 instance à l'aide de l'assistant de lancement d'instance de la console dans le guide de EC2 l'utilisateur Amazon.
2. Connectez-vous à votre instance. Pour plus d'informations, consultez Connect to your EC2 instance dans le guide de EC2 l'utilisateur Amazon.
3. Mettez à jour les packages installés et le cache du package sur votre instance.

   ```
   sudo yum update -y
   ```

4. Installez le package de Docker Community Edition le plus récent.

   ```
   sudo yum install docker
   ```

5. Lancez le service Docker.

   ```
   sudo service docker start
   ```

6. Ajoutez le `ec2-user` au groupe `docker` afin de pouvoir exécuter les commandes Docker sans utiliser le `sudo`.

   ```
   sudo usermod -a -G docker ec2-user
   ```

7. Déconnectez-vous et reconnectez-vous pour récupérer les nouvelles autorisations de groupe `docker`. Vous pouvez effectuer ces opérations en fermant votre fenêtre de terminal SSH actuelle et en vous reconnectant à votre instance dans une nouvelle fenêtre. Votre nouvelle session SSH disposera des autorisations de groupe `docker` appropriées.

8. Vérifiez que `ec2-user` peut exécuter les commandes Docker sans `sudo`.

```
docker info
```

> **Note**
>
> Dans certains cas, vous devrez peut-être redémarrer votre instance pour autoriser `ec2-user` à accéder au démon Docker. Essayez de redémarrer l'instance si vous voyez l'erreur suivante :
>
> ```
> Cannot connect to the Docker daemon. Is the docker daemon running on this
> host?
> ```

Créer une image Docker

Les définitions de tâches Amazon ECS utilisent des images de conteneurs pour lancer des conteneurs sur les instances de conteneur de vos clusters. Dans cette section, vous créez une image Docker d'une application Web simple, vous la testez sur votre système local ou sur une EC2 instance Amazon, puis vous envoyez l'image dans le registre de conteneurs Amazon ECR afin de pouvoir l'utiliser dans une définition de tâche Amazon ECS.

Créer une image Docker d'une application web simple

1. Créez un fichier, appelé `Dockerfile`. Un fichier Dockerfile est un manifeste qui décrit l'image de base à utiliser pour votre image Docker et ce que vous voulez installer et exécuter dessus. Pour en savoir plus sur les fichiers Dockerfile, consultez la référence Dockerfile.

```
touch Dockerfile
```

2. Modifiez le `Dockerfile` que vous venez de créer et ajoutez le contenu qui suit.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest

# Update installed packages and install Apache
RUN yum update -y && \
 yum install -y httpd

# Write hello world message
```

```
RUN echo 'Hello World!' > /var/www/html/index.html

# Configure Apache
RUN echo 'mkdir -p /var/run/httpd' >> /root/run_apache.sh && \
 echo 'mkdir -p /var/lock/httpd' >> /root/run_apache.sh && \
 echo '/usr/sbin/httpd -D FOREGROUND' >> /root/run_apache.sh && \
 chmod 755 /root/run_apache.sh

EXPOSE 80

CMD /root/run_apache.sh
```

Ce Dockerfile utilise l'image publique Amazon Linux 2023 hébergée sur Amazon ECR Public. Les instructions `RUN` mettent à jour les caches du package, installent certains packages logiciels pour le serveur web et écrivent ensuite le message « Hello World! » contenu à la racine du document des serveurs Web. L'`EXPOSE`instruction signifie que le port 80 du conteneur est celui qui écoute, et l'`CMD`instruction démarre le serveur Web.

3. Créez l'image Docker à partir de votre fichier Dockerfile.

> **Note**
>
> Certaines versions de Docker exigent le chemin d'accès complet à votre Dockerfile dans la commande suivante au lieu du chemin d'accès relatif indiqué ci-après.
> Si vous exécutez la commande sur un système ARM, tel qu'Apple Silicon, utilisez l'option --platform « --platform linux/amd64 ».

```
docker build -t hello-world .
```

4. Répertoriez l'image de votre conteneur.

```
docker images --filter reference=hello-world
```

Sortie :

```
REPOSITORY          TAG                 IMAGE ID            CREATED
 SIZE
hello-world         latest              e9ffedc8c286        4 minutes ago
 194MB
```

5. Exécutez la nouvelle image. L'option `-p 80:80` mappe le port exposé 80 du conteneur au port 80 du système hôte.

```
docker run -t -i -p 80:80 hello-world
```

> **Note**
>
> La sortie du serveur Web Apache est affichée dans la fenêtre du terminal. Vous pouvez ignorer le message « `Could not reliably determine the fully qualified domain name` ».

6. Ouvrez un navigateur et pointez vers le serveur qui exécute Docker et qui héberge votre conteneur.

   - Si vous utilisez une EC2 instance, il s'agit de la valeur DNS publique du serveur, qui est la même adresse que celle que vous utilisez pour vous connecter à l'instance via SSH. Assurez-vous que le groupe de sécurité de votre instance autorise le trafic entrant sur le port 80.
   - Si vous exécutez Docker localement, pointez votre navigateur vers http://localhost/.

   Vous devriez voir une page web avec « Hello, World! » .

7. Arrêtez le conteneur Docker en appuyant sur Ctrl + c.

Transmettre votre image à Amazon Elastic Container Registry

Amazon ECR est un service de registre d'images AWS géré. Vous pouvez utiliser la CLI Docker pour transmettre, extraire et gérer les images dans vos référentiels Amazon ECR. Pour en savoir plus sur les produits Amazon ECR, consulter les études de cas présentées par des clients et FAQs consulter les pages détaillées des produits Amazon Elastic Container Registry.

Pour étiqueter votre image et la transmettre à Amazon ECR

1. Créez un référentiel Amazon ECR pour stocker votre image `hello-world`. Notez la valeur `repositoryUri` dans la sortie.

   Remplacez`region`, par exemple Région AWS, par votre`us-east-1`.

```
aws ecr create-repository --repository-name hello-repository --region region
```

Sortie :

```
{
    "repository": {
        "registryId": "aws_account_id",
        "repositoryName": "hello-repository",
        "repositoryArn": "arn:aws:ecr:region:aws_account_id:repository/hello-
repository",
        "createdAt": 1505337806.0,
        "repositoryUri": "aws_account_id.dkr.ecr.region.amazonaws.com/hello-
repository"
    }
}
```

2. Étiquetez l'image `hello-world` avec la valeur `repositoryUri` de l'étape précédente.

```
docker tag hello-world aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository
```

3. Exécutez la commande aws ecr get-login-password. Spécifiez l'URI du registre sur lequel vous souhaitez vous authentifier. Pour plus d'informations, veuillez consulter Authentification de registre dans le Guide de l'utilisateur Amazon Elastic Container Registry.

```
aws ecr get-login-password --region region | docker login --username AWS --
password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

Sortie :

```
Login Succeeded
```

> **Important**
>
> Si vous recevez une erreur, installez la dernière version de l' AWS CLI ou effectuez une mise à niveau vers cette version. Pour plus d'informations, voir Installation ou mise à jour vers la dernière version du AWS CLI dans le guide de AWS Command Line Interface l'utilisateur.

4. Transférez l'image dans Amazon ECR avec la valeur `repositoryUri` de l'étape précédente.

```
docker push aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository
```

Nettoyage

Pour continuer à créer une définition de tâche Amazon ECS et à lancer une tâche avec votre image de conteneur, passez à la page Étapes suivantes. Une fois que vous avez fini de tester votre image Amazon ECR, vous pouvez supprimer le référentiel afin qu'aucun stockage d'images ne vous soit facturé.

```
aws ecr delete-repository --repository-name hello-repository --region region --force
```

Étapes suivantes

Vos définitions de tâches nécessitent un rôle d'exécution de tâche. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

Après avoir créé et transféré votre image de conteneur vers Amazon ECR, vous pouvez utiliser cette image dans une définition de tâche. Pour plus d'informations, consultez les étapes suivantes :

- the section called “Découvrez comment créer une tâche Linux pour le type de lancement Fargate”
- the section called “Découvrez comment créer une tâche Windows pour le type de lancement Fargate”
- Création d'une tâche Linux Amazon ECS pour le type de lancement Fargate avec AWS CLI

Découvrez comment créer une tâche Linux Amazon ECS pour le type de lancement Fargate

Amazon Elastic Container Service (Amazon ECS) est un service de gestion de conteneurs hautement évolutif et rapide, qui facilite l'exécution, l'arrêt et la gestion de vos conteneurs. Vous pouvez héberger vos conteneurs sur une infrastructure sans serveur gérée par Amazon ECS en lançant vos services ou vos tâches sur AWS Fargate. Pour plus d'informations sur Fargate, consultez. AWS Fargatepour Amazon ECS

Commencez à utiliser Amazon ECS AWS Fargate en utilisant le type de lancement Fargate pour vos tâches dans les régions où Amazon ECS prend en charge Fargate. AWS

Procédez comme suit pour démarrer avec Amazon ECS sur AWS Fargate.

Prérequis

Avant de commencer, suivez les étapes ci-dessous Configurer l'utilisation d'Amazon ECS et assurez-vous que votre utilisateur IAM dispose des autorisations spécifiées dans l'exemple de politique `AdministratorAccess` IAM.

La console tente de créer automatiquement le rôle IAM d'exécution de tâche, qui est requis pour les tâches Fargate. Pour garantir que la console peut créer ce rôle IAM, l'une des conditions suivantes doit être remplie :

- Votre utilisateur dispose d'un accès administrateur. Pour de plus amples informations, veuillez consulter Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur dispose des autorisations IAM nécessaires pour créer une fonction de service. Pour plus d'informations, voir Création d'un rôle pour déléguer des autorisations à un AWS service.
- Un utilisateur disposant d'un accès administrateur a créé manuellement le rôle d'exécution de tâche afin qu'il soit disponible sur le compte à utiliser. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

> **Important**
>
> Le groupe de sécurité que vous sélectionnez lors de la création d'un service avec votre définition de tâche doit avoir le port 80 ouvert pour le trafic entrant. Ajoutez les règles entrantes suivantes à votre groupe de sécurité. Pour plus d'informations sur la création d'un groupe de sécurité, consultez la section Créer un groupe de sécurité pour votre EC2 instance Amazon dans le guide de EC2 l'utilisateur Amazon.
>
> - Type : HTTP
> - Protocole : TCP
> - Plage de ports : 80
> - Source : n'importe où (`0.0.0.0/0`)

Étape 1 : créer le cluster

Créez un cluster qui utilise le VPC par défaut.

Avant de commencer, attribuez l'autorisation IAM adéquate. Pour de plus amples informations, veuillez consulter the section called “Exemples de clusters Amazon ECS”.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Cluster configuration (Configuration de cluster), pour Cluster name (Nom du cluster), saisissez un nom unique.

   Le nom peut contenir jusqu'à 255 lettres (minuscules et majuscules), des chiffres et des traits d'union.
6. (Facultatif) Pour activer Container Insights, développez Monitoring (Surveillance), puis activez Use Container Insights (Utiliser Container Insights).
7. (Facultatif) Pour vous aider à identifier votre cluster, développez Tags (balises), puis configurez vos balises.

   [Add a tag] Choisissez Add tag (Ajouter une balise) et procédez comme suit :

   - Pour Key (Clé), saisissez le nom de la clé.
   - Pour Value (Valeur), saisissez la valeur de clé.

   [Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de l'étiquette.
8. Choisissez Créer.

Étape 2 : Créer une définition de tâche

Une définition de tâche est similaire au modèle d'une application. Chaque fois que vous lancez une tâche dans Amazon ECS, vous spécifiez une définition de tâche. Le service connaît ensuite l'image Docker à utiliser pour les conteneurs, le nombre de conteneurs à utiliser dans la tâche et l'allocation des ressources pour chaque conteneur.

1. Dans le panneau de navigation, sélectionnez Task Definitions (Définition des tâches).
2. Choisissez Create new Task Definition (Créer une nouvelle définition de tâche), puis Create new revision with JSON (Créer une nouvelle révision avec JSON).

3. Copiez et collez l'exemple de définition de tâche suivant dans la zone, puis choisissez Save (Enregistrer).

```
{
    "family": "sample-fargate",
    "networkMode": "awsvpc",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": [
                "sh",
  "-c"
            ],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
            ]
        }
    ],
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "cpu": "256",
    "memory": "512"
}
```

4. Choisissez Créer.

Étape 3 : Créer le service

Créez un service à l'aide de la définition de tâche.

1. Dans le volet de navigation, choisissez Clusters, puis sélectionnez le cluster que vous avez créé dans Étape 1 : créer le cluster.
2. Sous l'onglet Services choisissez Create (Créer).
3. Sous Deployment configuration (Configuration du déploiement), spécifiez la manière dont votre application est déployée.
   a. Pour Task Definition (Définition de tâche), choisissez la définition de tâche que vous avez créé dans Étape 2 : Créer une définition de tâche.
   b. Pour Service name (Nom du service), saisissez un nom pour votre service.
   c. Pour Desired tasks (Tâches souhaitées), saisissez 1.
4. Sous Mise en réseau, vous pouvez créer un nouveau groupe de sécurité ou sélectionner un groupe de sécurité existant pour votre tâche. Assurez-vous que le groupe de sécurité que vous utilisez possède la règle entrante répertoriée sous Prérequis.
5. Choisissez Créer.

Étape 4 : Afficher votre service

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Choisissez le cluster sur lequel vous avez exécuté le service.
4. Dans l'onglet Services, sous Nom du service, choisissez le service que vous avez créé dans Étape 3 : Créer le service.
5. Cliquez sur l'onglet Tâches, puis sélectionnez la tâche dans votre service.
6. Sur la page des tâches, dans la section Configuration, sous IP publique, choisissez Adresse ouverte.

Étape 5 : nettoyer

Lorsque vous avez fini d'utiliser un cluster Amazon ECS, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources que vous n'utilisez pas.

Certaines ressources Amazon ECS, comme les tâches, les services, les clusters et les instances de conteneur, sont nettoyées à l'aide de la console Amazon ECS. Les autres ressources, telles que les EC2 instances Amazon, les équilibreurs de charge Elastic Load Balancing et les groupes Auto Scaling, doivent être nettoyées manuellement dans la EC2 console Amazon ou en supprimant la AWS CloudFormation pile qui les a créées.

1. Dans le panneau de navigation, choisissez Clusters.
2. Sur la page Clusters, sélectionnez le cluster que vous avez créé pour ce didacticiel.
3. Choisissez l'onglet Services.
4. Sélectionnez le service, puis choisissez Supprimer.
5. À l'invite de confirmation, saisissez delete (supprimer) puis choisissez Delete (Supprimer). Vous pouvez également utiliser l'`Force delete`option permettant à Amazon ECS de réduire le service en votre nom avant de le supprimer.

   Attendez que le service soit supprimé.
6. Choisissez Delete Cluster (Supprimer le cluster). À l'invite de confirmation, saisissez Supprimer ***`cluster-name`***, puis sélectionnez Supprimer. La suppression du cluster nettoie les ressources associées créées avec le cluster, notamment les groupes Auto Scaling ou les VPCs équilibreurs de charge.

Découvrez comment créer une tâche Windows Amazon ECS pour le type de lancement Fargate

Commencez à utiliser Amazon ECS AWS Fargate en utilisant le type de lancement Fargate pour vos tâches dans les régions où Amazon ECS prend en charge Fargate. AWS

Procédez comme suit pour démarrer avec Amazon ECS sur AWS Fargate.

Prérequis

Avant de commencer, suivez les étapes ci-dessous Configurer l'utilisation d'Amazon ECS et assurez-vous que votre utilisateur IAM dispose des autorisations spécifiées dans l'exemple de politique `AdministratorAccess` IAM.

La console tente de créer automatiquement le rôle IAM d'exécution de tâche, qui est requis pour les tâches Fargate. Pour garantir que la console peut créer ce rôle IAM, l'une des conditions suivantes doit être remplie :

- Votre utilisateur dispose d'un accès administrateur. Pour de plus amples informations, veuillez consulter Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur dispose des autorisations IAM nécessaires pour créer une fonction de service. Pour plus d'informations, voir Création d'un rôle pour déléguer des autorisations à un AWS service.
- Un utilisateur disposant d'un accès administrateur a créé manuellement le rôle d'exécution de tâche afin qu'il soit disponible sur le compte à utiliser. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

> **Important**
>
> Le groupe de sécurité que vous sélectionnez lors de la création d'un service avec votre définition de tâche doit avoir le port 80 ouvert pour le trafic entrant. Ajoutez les règles entrantes suivantes à votre groupe de sécurité. Pour plus d'informations sur la création d'un groupe de sécurité, consultez la section Créer un groupe de sécurité pour votre EC2 instance Amazon dans le guide de EC2 l'utilisateur Amazon.
>
> - Type : HTTP
> - Protocole : TCP
> - Plage de ports : 80
> - Source : n'importe où (`0.0.0.0/0`)

Étape 1 : créer un cluster

Vous pouvez créer un cluster appelé windows qui utilise le VPC par défaut.

Pour créer un cluster avec AWS Management Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Cluster configuration (Configuration de cluster), pour Cluster name (Nom du cluster), saisissez windows.

6. (Facultatif) Pour activer Container Insights, développez Monitoring (Surveillance), puis activez Use Container Insights (Utiliser Container Insights).
7. (Facultatif) Pour vous aider à identifier votre cluster, développez Tags (balises), puis configurez vos balises.

   [Add a tag] Choisissez Add tag (Ajouter une balise) et procédez comme suit :

   - Pour Key (Clé), saisissez le nom de la clé.
   - Pour Value (Valeur), saisissez la valeur de clé.

   [Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de l'étiquette.
8. Choisissez Créer.

Étape 2 : Enregistrer une définition de tâche Windows

Avant de pouvoir exécuter des conteneurs Windows dans votre cluster Amazon ECS, vous devez enregistrer une définition de tâche. L'exemple de définition de tâche suivant affiche une page web simple sur le port 8080 d'une instance de conteneur avec l'image de conteneur `mcr.microsoft.com/windows/servercore/iis`.

Pour enregistrer l'exemple de définition de tâche auprès du AWS Management Console

1. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
2. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
3. Copiez et collez l'exemple de définition de tâche suivant dans la zone, puis choisissez Save (Enregistrer).

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html
 -Type file -Value '<html> <head> <title>Amazon ECS Sample App</title>
 <style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
 <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p>'; C:\\ServiceMonitor.exe w3svc"],
            "entryPoint": [
```

```
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

4. Vérifiez vos informations et choisissez Create (Créer).

Étape 3 : Créer un service avec votre définition de tâche

Après avoir enregistré votre définition de tâche, vous pouvez placer des tâches dans votre cluster avec elle. La procédure suivante crée un service avec votre définition de tâche et place une tâche dans votre cluster.

Pour créer un service à partir de votre définition de tâche avec la console

1. Dans le volet de navigation, choisissez Clusters, puis sélectionnez le cluster que vous avez créé dans Étape 1 : créer un cluster.
2. Sous l'onglet Services choisissez Create (Créer).

3. Sous Deployment configuration (Configuration du déploiement), spécifiez la manière dont votre application est déployée.
   a. Pour Task Definition (Définition de tâche), choisissez la définition de tâche que vous avez créé dans Étape 2 : Enregistrer une définition de tâche Windows.
   b. Pour Service name (Nom du service), saisissez un nom pour votre service.
   c. Pour Desired tasks (Tâches souhaitées), saisissez 1.
4. Sous Mise en réseau, vous pouvez créer un groupe de sécurité ou sélectionner un groupe de sécurité existant. Assurez-vous que le groupe de sécurité que vous utilisez possède la règle entrante répertoriée sous Prérequis.
5. Choisissez Créer.

Étape 4 : Afficher votre service

Lorsque votre service a lancé une tâche dans votre cluster, vous pouvez afficher le service et ouvrir la page de test IIS dans un navigateur pour vérifier que le conteneur est en cours d'exécution.

> **Note**
>
> Cette opération peut prendre jusqu'à 15 minutes pour que votre instance de conteneur télécharge et extrait les couches de base du conteneur Windows.

Pour afficher votre service

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Choisissez le cluster sur lequel vous avez exécuté le service.
4. Dans l'onglet Services, sous Nom du service, choisissez le service que vous avez créé dans Étape 3 : Créer un service avec votre définition de tâche.
5. Cliquez sur l'onglet Tâches, puis sélectionnez la tâche dans votre service.
6. Sur la page des tâches, dans la section Configuration, sous IP publique, choisissez Adresse ouverte.

Étape 5 : Nettoyer

Lorsque vous avez fini d'utiliser un cluster Amazon ECS, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources que vous n'utilisez pas.

Certaines ressources Amazon ECS, comme les tâches, les services, les clusters et les instances de conteneur, sont nettoyées à l'aide de la console Amazon ECS. Les autres ressources, telles que les EC2 instances Amazon, les équilibreurs de charge Elastic Load Balancing et les groupes Auto Scaling, doivent être nettoyées manuellement dans la EC2 console Amazon ou en supprimant la AWS CloudFormation pile qui les a créées.

1. Dans le panneau de navigation, choisissez Clusters.
2. Sur la page Clusters, sélectionnez le cluster que vous avez créé pour ce didacticiel.
3. Choisissez l'onglet Services.
4. Sélectionnez le service, puis choisissez Supprimer.
5. À l'invite de confirmation, saisissez delete (supprimer) puis choisissez Delete (Supprimer).

   Attendez que le service soit supprimé.
6. Choisissez Delete Cluster (Supprimer le cluster). À l'invite de confirmation, saisissez Supprimer ***`cluster-name`***, puis sélectionnez Supprimer. La suppression du cluster nettoie les ressources associées créées avec le cluster, notamment les groupes Auto Scaling ou les VPCs équilibreurs de charge.

Découvrez comment créer une tâche Windows Amazon ECS pour le type de EC2 lancement

Commencez à utiliser Amazon ECS en utilisant le type de EC2 lancement en enregistrant une définition de tâche, en créant un cluster et en créant un service dans la console.

Procédez comme suit pour démarrer avec Amazon ECS en utilisant le type de EC2 lancement.

Prérequis

Avant de commencer, suivez les étapes ci-dessous Configurer l'utilisation d'Amazon ECS et assurez-vous que votre utilisateur IAM dispose des autorisations spécifiées dans l'exemple de politique `AdministratorAccess` IAM.

La console tente de créer automatiquement le rôle IAM d'exécution de tâche, qui est requis pour les tâches Fargate. Pour garantir que la console peut créer ce rôle IAM, l'une des conditions suivantes doit être remplie :

- Votre utilisateur dispose d'un accès administrateur. Pour de plus amples informations, veuillez consulter Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur dispose des autorisations IAM nécessaires pour créer une fonction de service. Pour plus d'informations, voir Création d'un rôle pour déléguer des autorisations à un AWS service.
- Un utilisateur disposant d'un accès administrateur a créé manuellement le rôle d'exécution de tâche afin qu'il soit disponible sur le compte à utiliser. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

> **Important**
>
> Le groupe de sécurité que vous sélectionnez lors de la création d'un service avec votre définition de tâche doit avoir le port 80 ouvert pour le trafic entrant. Ajoutez les règles entrantes suivantes à votre groupe de sécurité. Pour plus d'informations sur la création d'un groupe de sécurité, consultez la section Créer un groupe de sécurité pour votre EC2 instance Amazon dans le guide de EC2 l'utilisateur Amazon.
>
> - Type : HTTP
> - Protocole : TCP
> - Plage de ports : 80
> - Source : n'importe où (`0.0.0.0/0`)

Étape 1 : créer un cluster

Un cluster Amazon ECS est un regroupement logique de tâches, de services et d'instances de conteneur.

Les étapes suivantes vous expliquent comment créer un cluster dans lequel une EC2 instance Amazon est enregistrée, ce qui nous permettra d'exécuter une tâche dessus. Si un champ spécifique n'est pas mentionné, conservez les valeurs par défaut utilisées par la console.

Pour créer un nouveau cluster (console Amazon ECS)

Avant de commencer, attribuez l'autorisation IAM adéquate. Pour de plus amples informations, veuillez consulter the section called “Exemples de clusters Amazon ECS”.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Cluster configuration (Configuration de cluster), pour Cluster name (Nom du cluster), saisissez un nom unique.

   Le nom peut contenir jusqu'à 255 lettres (minuscules et majuscules), des chiffres et des traits d'union.
6. (Facultatif) Pour modifier le VPC et les sous-réseaux d'où vos tâches et services se lancent, sous Networking (Réseaux), effectuez l'une des opérations suivantes :
   - Pour supprimer un sous-réseau, sous Subnets (Sous-réseaux), choisissez X pour chaque sous-réseau que vous souhaitez supprimer.
   - Pour passer à un VPC autre celui par défaut, sous VPC, choisissez un VPC existant, puis sous Subnets (Sous-réseaux), sélectionnez chaque sous-réseau.
7. Pour ajouter des EC2 instances Amazon à votre cluster, développez l'infrastructure, puis sélectionnez EC2les instances Amazon. Ensuite, configurez le groupe Auto Scaling qui agit en tant que fournisseur de capacité :
   a. Pour utiliser un groupe Auto Scaling existant, depuis Auto Scaling group (ASG) (Groupe Auto Scaling [ASG]), sélectionnez le groupe.
   b. Pour créer un groupe Auto Scaling, depuis Auto Scaling group (ASG) (Groupe Auto Scaling [ASG]), sélectionnez Create new group (Créer un nouveau groupe), puis fournissez les détails suivants sur le groupe :
      - Pour Operating system/Architecture (Système d'exploitation/architecture), choisissez l'AMI optimisée pour Amazon ECS pour les instances de groupe Auto Scaling.
      - EC2 Par type d'instance, choisissez le type d'instance pour vos charges de travail. Pour plus d'informations sur les différents types d' EC2 instances, consultez Amazon Instances.

La mise à l'échelle gérée fonctionne mieux si votre groupe Auto Scaling utilise les mêmes types d'instance ou des types d'instance similaires.

- Pour SSH key pair (Paire de clés SSH), choisissez la paire qui prouve votre identité lorsque vous connectez à l'instance.
- Pour Capacity (Capacité), saisissez le nombre minimum et le nombre maximum d'instances à lancer dans le groupe Auto Scaling. EC2 Les instances Amazon entraînent des coûts tant qu'elles existent dans vos AWS ressources. Pour plus d'informations, consultez Amazon EC2 Pricing.

8. (Facultatif) Pour activer Container Insights, développez Monitoring (Surveillance), puis activez Use Container Insights (Utiliser Container Insights).
9. (Facultatif) Pour gérer les identifications de cluster, développez Tags (Identifications), puis effectuez l'une des opérations suivantes :

   [Add a tag] Choisissez Add tag (Ajouter une étiquette) et procédez comme suit :

   - Pour Key (Clé), saisissez le nom de la clé.
   - Pour Value (Valeur), saisissez la valeur de clé.

   [Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de l'étiquette.
10. Choisissez Créer.

Étape 2 : Enregistrer une définition de tâche

Pour enregistrer l'exemple de définition de tâche auprès du AWS Management Console

1. Dans le panneau de navigation, sélectionnez Task Definitions (Définition des tâches).
2. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
3. Copiez et collez l'exemple de définition de tâche suivant dans la zone, puis choisissez Enregistrer.

```
{
    "containerDefinitions": [
        {
```

```
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html
 -Type file -Value '<html> <head> <title>Amazon ECS Sample App</title>
 <style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
 <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p>'; C:\\ServiceMonitor.exe w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 443,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["EC2"]
}
```

4. Vérifiez vos informations et choisissez Create (Créer).

Étape 3 : Créer un service

Un service Amazon ECS service vous aide à exécuter et à gérer simultanément un nombre spécifié d'instances d'une définition de tâche dans un cluster Amazon ECS. Si l'une de vos tâches est défaillante ou s'arrête pour une raison quelconque, le planificateur Amazon ECS service lance une autre instance de votre définition de tâche pour la remplacer et maintenir le nombre souhaité

de tâches dans le service. Pour de plus amples informations sur les services, consultez Services Amazon ECS.

Pour créer un service

1. Dans le panneau de navigation, choisissez Clusters.
2. Sélectionnez le cluster que vous avez créé dans Étape 1 : créer un cluster.
3. Dans l'onglet Services choisissez Create (Créer).
4. Dans la section Environment (Environnement), procédez comme suit :
   a. Pour les Compute options (Options de calcul), choisissez Launch type (Type de lancement).
   b. Pour le type de lancement, sélectionnez EC2
5. Dans la section Deployment configuration (Configuration de déploiement), effectuez les opérations suivantes :
   a. Pour Family (Famille), choisissez la définition de tâche que vous avez créé dans Étape 2 : Enregistrer une définition de tâche.
   b. Pour Service name (Nom du service), saisissez un nom pour votre service.
   c. Pour Desired tasks (Tâches souhaitées), saisissez 1.
6. Vérifiez les options et choisissez Create (Créer).
7. Choisissez View service (Afficher le service) pour vérifier votre service.

Étape 4 : Afficher votre service

Le service est une application web qui vous permet d'afficher ses conteneurs à l'aide d'un navigateur web.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Choisissez le cluster sur lequel vous avez exécuté le service.
4. Dans l'onglet Services, sous Nom du service, choisissez le service que vous avez créé dans Étape 3 : Créer un service.
5. Cliquez sur l'onglet Tâches, puis sélectionnez la tâche dans votre service.
6. Sur la page des tâches, dans la section Configuration, sous IP publique, choisissez Adresse ouverte. La capture d'écran ci-dessous est le résultat attendu.

Étape 5 : Nettoyer

Lorsque vous avez fini d'utiliser un cluster Amazon ECS, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources que vous n'utilisez pas.

Certaines ressources Amazon ECS, comme les tâches, les services, les clusters et les instances de conteneur, sont nettoyées à l'aide de la console Amazon ECS. Les autres ressources, telles que les EC2 instances Amazon, les équilibreurs de charge Elastic Load Balancing et les groupes Auto Scaling, doivent être nettoyées manuellement dans la EC2 console Amazon ou en supprimant la AWS CloudFormation pile qui les a créées.

1. Dans le panneau de navigation, choisissez Clusters.
2. Sur la page Clusters, sélectionnez le cluster que vous avez créé pour ce didacticiel.
3. Choisissez l'onglet Services.
4. Sélectionnez le service, puis choisissez Supprimer.
5. À l'invite de confirmation, saisissez delete (supprimer) puis choisissez Delete (Supprimer).

   Attendez que le service soit supprimé.
6. Choisissez Delete Cluster (Supprimer le cluster). À l'invite de confirmation, saisissez Supprimer ***cluster-name***, puis sélectionnez Supprimer. La suppression du cluster nettoie les ressources associées créées avec le cluster, notamment les groupes Auto Scaling ou les VPCs équilibreurs de charge.

Création de ressources Amazon ECS à l'aide du AWS CDK

AWS Cloud Development Kit (AWS CDK) Il s'agit d'un framework Infrastructure-as-Code (IAC) que vous pouvez utiliser pour définir l'infrastructure AWS cloud en utilisant le langage de programmation de votre choix. Pour définir votre propre infrastructure de cloud, écrivez d'abord une application (dans l'un des langages pris en charge par le CDK) contenant une ou plusieurs piles. Ensuite, vous le synthétisez dans un AWS CloudFormation modèle et déployez vos ressources sur votre Compte AWS. Suivez les étapes décrites dans cette rubrique pour déployer un serveur Web conteneurisé avec Amazon Elastic Container Service (Amazon ECS) et on Fargate. AWS CDK

La bibliothèque AWS Construct, incluse dans le CDK, fournit des modules que vous pouvez utiliser pour modéliser les ressources Services AWS fournies. Pour les services populaires, la bibliothèque fournit des constructions organisées avec des valeurs par défaut intelligentes et les bonnes pratiques. L'un de ces modules, en particulier `aws-ecs-patterns`, fournit des abstractions de haut niveau que vous pouvez utiliser pour définir votre service conteneurisé et toutes les ressources de support nécessaires en quelques lignes de code.

Cette rubrique utilise la construction `ApplicationLoadBalancedFargateService`. Cette construction déploie un Amazon ECS service sur Fargate derrière un Application Load Balancer. Le `aws-ecs-patterns` module inclut également des constructions qui utilisent un équilibreur de charge réseau et s'exécutent sur Amazon. EC2

Avant de commencer cette tâche, configurez votre environnement de AWS CDK développement et installez-le AWS CDK en exécutant la commande suivante. Pour obtenir des instructions sur la configuration de votre environnement de AWS CDK développement, consultez Getting Started with the AWS CDK - Prérequis.

```
npm install -g aws-cdk
```

> **Note**
>
> Ces instructions supposent que vous utilisez la AWS CDK version 2.

Rubriques

- Étape 1 : Configurez votre AWS CDK projet
- Étape 2 : utilisez le AWS CDK pour définir un serveur Web conteneurisé sur Fargate

- Étape 3 : Tester le service web
- Étape 4 : nettoyer
- Étapes suivantes

Étape 1 : Configurez votre AWS CDK projet

Créez un répertoire pour votre nouvelle AWS CDK application et initialisez le projet.

TypeScript

```
mkdir hello-ecs
cd hello-ecs
cdk init --language typescript
```

JavaScript

```
mkdir hello-ecs
cd hello-ecs
cdk init --language javascript
```

Python

```
mkdir hello-ecs
cd hello-ecs
cdk init --language python
```

Une fois le projet démarré, activez l'environnement virtuel du projet et installez ses dépendances AWS CDK de base.

```
source .venv/bin/activate
python -m pip install -r requirements.txt
```

Java

```
mkdir hello-ecs
cd hello-ecs
cdk init --language java
```

Importez ce projet Maven dans votre IDE Java. Par exemple, dans Eclipse, utilisez Fichier >Importer >Maven >Projets Maven existants.

C#

```
mkdir hello-ecs
cd hello-ecs
cdk init --language csharp
```

Go

```
mkdir hello-ecs
cd hello-ecs
cdk init --language go
```

> **Note**
>
> Le modèle AWS CDK d'application utilise le nom du répertoire du projet pour générer des noms pour les classes et les fichiers sources. Dans cet exemple, le répertoire est nommé `hello-ecs`. Si vous utilisez un nom répertoire différent, votre application ne correspondra pas à ces instructions.

AWS CDK v2 inclut des constructions stables pour tous Services AWS dans un seul package appelé`aws-cdk-lib`. Ce package est installé en tant que dépendance lorsque vous initialisez le projet. Lorsque vous travaillez avec certains langages de programmation, le package est installé lorsque vous créez le projet pour la première fois. Cette rubrique explique comment utiliser une construction Amazon ECS Patterns, qui fournit des abstractions de haut niveau pour travailler avec Amazon ECS. Ce module s'appuie sur les constructions Amazon ECS et d'autres constructions pour allouer les ressources nécessaires à votre application Amazon ECS.

Les noms que vous utilisez pour importer ces bibliothèques dans votre application CDK peuvent différer légèrement selon le langage de programmation que vous utilisez. À titre de référence, les noms suivants sont utilisés dans chaque langage de programmation CDK pris en charge.

TypeScript

```
aws-cdk-lib/aws-ecs
aws-cdk-lib/aws-ecs-patterns
```

JavaScript

```
aws-cdk-lib/aws-ecs
aws-cdk-lib/aws-ecs-patterns
```

Python

```
aws_cdk.aws_ecs
aws_cdk.aws_ecs_patterns
```

Java

```
software.amazon.awscdk.services.ecs
software.amazon.awscdk.services.ecs.patterns
```

C#

```
Amazon.CDK.AWS.ECS
Amazon.CDK.AWS.ECS.Patterns
```

Go

```
github.com/aws/aws-cdk-go/awscdk/v2/awsecs
github.com/aws/aws-cdk-go/awscdk/v2/awsecspatterns
```

Étape 2 : utilisez le AWS CDK pour définir un serveur Web conteneurisé sur Fargate

Utilisez l'image du conteneur `amazon-ecs-sample`. Cette image contient une application Web PHP qui s'exécute sur Ngingx.

Dans le AWS CDK projet que vous avez créé, modifiez le fichier contenant la définition de la pile pour qu'il ressemble à l'un des exemples suivants.

> **Note**
>
> Une pile est une unité de déploiement. Toutes les ressources doivent se trouver dans une pile et toutes les ressources d'une pile sont déployées en même temps. Si une ressource ne parvient pas à se déployer, toutes les autres ressources déjà déployées sont annulées. Une

AWS CDK application peut contenir plusieurs piles, et les ressources d'une pile peuvent faire référence aux ressources d'une autre pile.

TypeScript

Mise à jour `lib/hello-ecs-stack.ts` de sorte qu'il ressemble à ce qui suit.

```
import * as cdk from 'aws-cdk-lib';
import { Construct } from 'constructs';
import * as ecs from 'aws-cdk-lib/aws-ecs';
import * as ecsp from 'aws-cdk-lib/aws-ecs-patterns';

export class HelloEcsStack extends cdk.Stack {
  constructor(scope: Construct, id: string, props?: cdk.StackProps) {
    super(scope, id, props);

    new ecsp.ApplicationLoadBalancedFargateService(this, 'MyWebServer', {
      taskImageOptions: {
        image: ecs.ContainerImage.fromRegistry('public.ecr.aws/ecs-sample-image/
amazon-ecs-sample:latest'),
      },
      publicLoadBalancer: true
    });
  }
}
```

JavaScript

Mise à jour `lib/hello-ecs-stack.js` de sorte qu'il ressemble à ce qui suit.

```
const cdk = require('aws-cdk-lib');
const { Construct } = require('constructs');
const ecs = require('aws-cdk-lib/aws-ecs');
const ecsp = require('aws-cdk-lib/aws-ecs-patterns');

class HelloEcsStack extends cdk.Stack {
  constructor(scope = Construct, id = string, props = cdk.StackProps) {
    super(scope, id, props);

    new ecsp.ApplicationLoadBalancedFargateService(this, 'MyWebServer', {
      taskImageOptions: {
        image: ecs.ContainerImage.fromRegistry('amazon/amazon-ecs-sample'),
```

```
      },
      publicLoadBalancer: true
    });
  }
}

module.exports = { HelloEcsStack }
```

Python

Mise à jour `hello-ecs/hello_ecs_stack.py` de sorte qu'il ressemble à ce qui suit.

```
import aws_cdk as cdk
from constructs import Construct

import aws_cdk.aws_ecs as ecs
import aws_cdk.aws_ecs_patterns as ecsp

class HelloEcsStack(cdk.Stack):

    def __init__(self, scope: Construct, construct_id: str, **kwargs) -> None:
        super().__init__(scope, construct_id, **kwargs)

        ecsp.ApplicationLoadBalancedFargateService(self, "MyWebServer",
            task_image_options=ecsp.ApplicationLoadBalancedTaskImageOptions(
                image=ecs.ContainerImage.from_registry("amazon/amazon-ecs-sample")),
            public_load_balancer=True
        )
```

Java

Mise à jour `src/main/java/com.myorg/HelloEcsStack.java` de sorte qu'il ressemble à ce qui suit.

```
package com.myorg;

import software.constructs.Construct;
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;

import software.amazon.awscdk.services.ecs.ContainerImage;
import
 software.amazon.awscdk.services.ecs.patterns.ApplicationLoadBalancedFargateService;
```

```
import
 software.amazon.awscdk.services.ecs.patterns.ApplicationLoadBalancedTaskImageOptions;

public class HelloEcsStack extends Stack {
    public HelloEcsStack(final Construct scope, final String id) {
        this(scope, id, null);
    }

    public HelloEcsStack(final Construct scope, final String id, final StackProps
 props) {
        super(scope, id, props);

        ApplicationLoadBalancedFargateService.Builder.create(this, "MyWebServer")
         .taskImageOptions(ApplicationLoadBalancedTaskImageOptions.builder()
           .image(ContainerImage.fromRegistry("amazon/amazon-ecs-sample"))
           .build())
         .publicLoadBalancer(true)
         .build();
    }
}
```

C#

Mise à jour `src/HelloEcs/HelloEcsStack.cs` de sorte qu'il ressemble à ce qui suit.

```
using Amazon.CDK;
using Constructs;
using Amazon.CDK.AWS.ECS;
using Amazon.CDK.AWS.ECS.Patterns;
namespace HelloEcs
{
    public class HelloEcsStack : Stack
    {
        internal HelloEcsStack(Construct scope, string id, IStackProps props =
 null) : base(scope, id, props)
        {
            new ApplicationLoadBalancedFargateService(this, "MyWebServer",
                new ApplicationLoadBalancedFargateServiceProps
                {
                    TaskImageOptions = new ApplicationLoadBalancedTaskImageOptions
                    {
                        Image = ContainerImage.FromRegistry("amazon/amazon-ecs-
sample")
                    },
```

```
                    PublicLoadBalancer = true
                });
        }
    }
}
```

Go

Mise à jour `hello-ecs.go` de sorte qu'il ressemble à ce qui suit.

```
package main

import (
 "github.com/aws/aws-cdk-go/awscdk/v2"
 // "github.com/aws/aws-cdk-go/awscdk/v2/awssqs"
 "github.com/aws/aws-cdk-go/awscdk/v2/awsecs"
 "github.com/aws/aws-cdk-go/awscdk/v2/awsecspatterns"
 "github.com/aws/constructs-go/constructs/v10"
 "github.com/aws/jsii-runtime-go"
)

type HelloEcsStackProps struct {
 awscdk.StackProps
}

func NewHelloEcsStack(scope constructs.Construct, id string, props
 *HelloEcsStackProps) awscdk.Stack {
 var sprops awscdk.StackProps
 if props != nil {
  sprops = props.StackProps
 }
 stack := awscdk.NewStack(scope, &id, &sprops)

 // The code that defines your stack goes here

 // example resource
 // queue := awssqs.NewQueue(stack, jsii.String("HelloEcsQueue"),
 &awssqs.QueueProps{
 //  VisibilityTimeout: awscdk.Duration_Seconds(jsii.Number(300)),
 // })
 res := awsecspatterns.NewApplicationLoadBalancedFargateService(stack,
 jsii.String("MyWebServer"),
  &awsecspatterns.ApplicationLoadBalancedFargateServiceProps{
   TaskImageOptions: &awsecspatterns.ApplicationLoadBalancedTaskImageOptions{
```

```
    Image: awsecs.ContainerImage_FromRegistry(jsii.String("amazon/amazon-ecs-
sample"), &awsecs.RepositoryImageProps{}),
   },
  },
 )
 awscdk.NewCfnOutput(stack, jsii.String("LoadBalancerDNS"),
 &awscdk.CfnOutputProps{Value: res.LoadBalancer().LoadBalancerDnsName()})

 return stack
}

func main() {
 defer jsii.Close()

 app := awscdk.NewApp(nil)

 NewHelloEcsStack(app, "HelloEcsStack", &HelloEcsStackProps{
  awscdk.StackProps{
   Env: env(),
  },
 })

 app.Synth(nil)
}

// env determines the AWS environment (account+region) in which our stack is to
// be deployed. For more information see: https://docs.aws.amazon.com/cdk/latest/
guide/environments.html
func env() *awscdk.Environment {
 // If unspecified, this stack will be "environment-agnostic".
 // Account/Region-dependent features and context lookups will not work, but a
 // single synthesized template can be deployed anywhere.
 //---------------------------------------------------------------------------
 return nil

 // Uncomment if you know exactly what account and region you want to deploy
 // the stack to. This is the recommendation for production stacks.
 //---------------------------------------------------------------------------
 // return &awscdk.Environment{
 //  Account: jsii.String("123456789012"),
 //  Region:  jsii.String("us-east-1"),
 // }

 // Uncomment to specialize this stack for the AWS Account and Region that are
```

```
 // implied by the current CLI configuration. This is recommended for dev
 // stacks.
 //---------------------------------------------------------------------------
 // return &awscdk.Environment{
 //  Account: jsii.String(os.Getenv("CDK_DEFAULT_ACCOUNT")),
 //  Region:  jsii.String(os.Getenv("CDK_DEFAULT_REGION")),
 // }
}
```

Le court extrait précédent inclut les éléments suivants :

- Le nom logique du service : `MyWebServer`.
- L'image du conteneur obtenue à partir de la galerie publique Amazon ECR :`amazon/amazon-ecs-sample`.
- D'autres informations pertinentes, telles que le fait que l'équilibreur de charge a une adresse publique et est accessible à partir d'Internet.

Cela AWS CDK créera toutes les ressources nécessaires au déploiement du serveur Web, y compris les ressources suivantes. Ces ressources ont été omises dans cet exemple.

- Cluster Amazon ECS
- Amazon VPC et instances Amazon EC2
- Groupe Auto Scaling
- Application Load Balancer
- Rôles et politiques IAM

Certaines ressources allouées automatiquement sont partagées par tous les Amazon ECS services définis dans la pile.

Enregistrez le fichier source, puis exécutez la commande `cdk synth` dans le répertoire principal de l'application. AWS CDK Exécute l'application et synthétise un AWS CloudFormation modèle à partir de celle-ci, puis affiche le modèle. Le modèle est un fichier YAML d'environ 600 lignes. Le début du fichier est illustré ici. Votre modèle peut être différent de cet exemple.

```
Resources:
  MyWebServerLB3B5FD3AB:
```

```
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      LoadBalancerAttributes:
        - Key: deletion_protection.enabled
          Value: "false"
      Scheme: internet-facing
      SecurityGroups:
        - Fn::GetAtt:
            - MyWebServerLBSecurityGroup01B285AA
            - GroupId
      Subnets:
        - Ref: EcsDefaultClusterMnL3mNNYNVpcPublicSubnet1Subnet3C273B99
        - Ref: EcsDefaultClusterMnL3mNNYNVpcPublicSubnet2Subnet95FF715A
      Type: application
    DependsOn:
      - EcsDefaultClusterMnL3mNNYNVpcPublicSubnet1DefaultRouteFF4E2178
      - EcsDefaultClusterMnL3mNNYNVpcPublicSubnet2DefaultRouteB1375520
    Metadata:
      aws:cdk:path: HelloEcsStack/MyWebServer/LB/Resource
  MyWebServerLBSecurityGroup01B285AA:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Automatically created Security Group for ELB
 HelloEcsStackMyWebServerLB06757F57
      SecurityGroupIngress:
        - CidrIp: 0.0.0.0/0
          Description: Allow from anyone on port 80
          FromPort: 80
          IpProtocol: tcp
          ToPort: 80
      VpcId:
        Ref: EcsDefaultClusterMnL3mNNYNVpc7788A521
    Metadata:
      aws:cdk:path: HelloEcsStack/MyWebServer/LB/SecurityGroup/Resource
# and so on for another few hundred lines
```

Pour déployer le service dans votre Compte AWS, exécutez la `cdk deploy` commande dans le répertoire principal de votre application. Il vous est demandé d'approuver les politiques IAM AWS CDK générées.

Le déploiement prend plusieurs minutes au cours desquelles plusieurs ressources sont créées. AWS CDK Les dernières lignes de la sortie du déploiement incluent le nom d'hôte public de l'équilibreur de charge et l'URL de votre nouveau serveur web. Ce sont les suivants.

```
Outputs:
HelloEcsStack.MyWebServerLoadBalancerDNSXXXXXXX = Hello-MyWeb-ZZZZZZZZZZZZZ-
ZZZZZZZZZZ.us-west-2.elb.amazonaws.com
HelloEcsStack.MyWebServerServiceURLYYYYYYYY = http://Hello-MyWeb-ZZZZZZZZZZZZZ-
ZZZZZZZZZZ.us-west-2.elb.amazonaws.com
```

Étape 3 : Tester le service web

Copiez l'URL de la sortie de déploiement et collez-la dans votre navigateur web. Le message de bienvenue suivant provenant du serveur web s'affiche.

Étape 4 : nettoyer

Une fois que vous avez terminé d'utiliser le serveur web, arrêtez le service à l'aide du CDK en exécutant la commande `cdk destroy` dans le répertoire principal de votre application. Cela vous évite de subir des frais imprévus à l'future.

Étapes suivantes

Pour en savoir plus sur le développement d'une AWS infrastructure à l'aide de AWS CDK, consultez le guide du AWS CDK développeur.

Pour plus d'informations sur l'écriture d' AWS CDK applications dans la langue de votre choix, consultez les rubriques suivantes :

TypeScript

- Travailler avec le AWS CDK in TypeScript

JavaScript

Travailler avec le AWS CDK in JavaScript

Python

Travailler avec le AWS CDK en Python

Java

Travailler avec le AWS CDK en Java

C#

Travailler avec le AWS CDK en C#

Go

Travailler avec AWS CDK In Go

Pour plus d'informations sur les modules AWS Construct Library utilisés dans cette rubrique, consultez les aperçus des références AWS CDK d'API suivants.

- aws-ecs
- aws-ecs-patterns

Création de ressources Amazon ECS à l'aide de l'interface de ligne de commande AWS Copilot

Les commandes de l'interface de ligne de commande (CLI) AWS Copilot simplifient la création, le lancement et l'exploitation d'applications conteneurisées prêtes pour la production sur Amazon ECS à partir d'un environnement de développement local. La AWS CLI Copilot s'aligne sur les flux de travail des développeurs qui prennent en charge les meilleures pratiques en matière d'applications modernes, qu'il s'agisse de l'utilisation de l'infrastructure sous forme de code ou de la création d'un CI/CD pipeline provisionné pour le compte d'un utilisateur. Utilisez la AWS CLI Copilot dans le cadre de votre cycle quotidien de développement et de test comme alternative au. AWS Management Console

AWS Copilot est actuellement compatible avec les systèmes Linux, macOS et Windows. Pour plus d'informations sur la dernière version de la AWS CLI Copilot, consultez la section Releases.

> **Note**
>
> Le code source de la AWS CLI Copilot est disponible sur. GitHub Nous vous conseillons de soumettre les problèmes et d'extraire les requêtes pour les modifications que vous souhaitez inclure. Toutefois, Amazon Web Services ne prend actuellement pas en charge l'exécution de copies modifiées du code d' AWS Copilot. Signalez les problèmes liés à AWS Copilot en communiquant avec nous sur Gitter ou sur Gitter, GitHubou vous pouvez signaler des problèmes, faire part de vos commentaires et signaler des bogues.
> Pour plus d'informations sur le développement de Copilot, voir Quel est le futur de Copilot ? .

Rubriques

- Installation de la CLI AWS Copilot
- Déploiement d'un exemple d'application Amazon ECS à l'aide de la CLI AWS Copilot

Une documentation supplémentaire pour la AWS CLI Copilot est disponible sur le site Web de Copilot AWS

Installation de la CLI AWS Copilot

Vous pouvez installer la AWS CLI Copilot en utilisant Homebrew ou en téléchargeant manuellement le binaire en suivant les étapes suivantes.

Utiliser Homebrew

La commande suivante est utilisée pour installer la AWS CLI Copilot sur votre système macOS ou Linux à l'aide de Homebrew. Avant l'installation, Homebrew doit être installé. Pour plus d'informations, consultez Homebrew.

```
brew install aws/tap/copilot-cli
```

Télécharger le binaire

Comme alternative à Homebrew, vous pouvez installer manuellement la CLI AWS Copilot sur votre système macOS, Windows ou Linux. Utilisez la commande suivante pour votre système d'exploitation pour télécharger le fichier binaire. Les exemples de macOS et Linux incluent également des commandes qui appliquent des autorisations d'exécution au binaire et répertorient le menu d'aide pour vérifier que l'installation fonctionne.

macOS

Pour Mac OS :

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-darwin \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Pour les systèmes macOS ARM :

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-darwin-arm64 \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Linux

Pour les systèmes Linux x86 (64 bits) :

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-linux \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Pour les systèmes ARM Linux :

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-linux-arm64 \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Windows

Exécutez la commande suivante à l'aide de PowerShell :

```
New-Item -Path 'C:\copilot' -ItemType directory; `
  Invoke-WebRequest -OutFile 'C:\copilot\copilot.exe' https://github.com/aws/
copilot-cli/releases/latest/download/copilot-windows.exe
```

(Facultatif) Vérifier la CLI AWS Copilot installée manuellement à l'aide des signatures PGP

Les exécutables de la AWS CLI Copilot sont signés cryptographiquement à l'aide de signatures PGP. Les signatures PGP peuvent être utilisées pour vérifier la validité de l'exécutable de la CLI AWS Copilot. Utilisez les étapes suivantes pour vérifier les signatures à l'aide de l'outil GnuPG.

1. Téléchargez et installez GnuPG. Pour plus d'informations, consultez le site web GnuPG.

   macOS

   Nous vous recommandons d'utiliser Homebrew. Installez Homebrew en suivant les instructions fournies sur leur site web. Pour plus d'informations, consultez Homebrew. Lorsqu'Homebrew est installé, utilisez la commande suivante à partir de votre terminal macOS.

   ```
   brew install gnupg
   ```

   Linux

   Installez gpg à l'aide du gestionnaire de packages sur votre version de Linux.

   Windows

   Téléchargez le programme d'installation Windows simple à partir du site web GnuPG et installez-le en tant qu'administrateur. Après avoir installé GnuPG, fermez puis rouvrez l'administrateur. PowerShell

   Pour plus d'informations, consultez Téléchargement de GnuPG.

2. Vérifiez que le chemin GnuPG est ajouté à votre chemin d'environnement.

   macOS

   ```
   echo $PATH
   ```

   Si vous ne voyez pas le chemin GnuPG dans la sortie, exécutez la commande suivante pour l'ajouter au chemin.

   ```
   PATH=$PATH:<path to GnuPG executable files>
   ```

Linux

```
echo $PATH
```

Si vous ne voyez pas le chemin GnuPG dans la sortie, exécutez la commande suivante pour l'ajouter au chemin.

```
export PATH=$PATH:<path to GnuPG executable files>
```

Windows

```
Write-Output $Env:PATH
```

Si vous ne voyez pas le chemin GnuPG dans la sortie, exécutez la commande suivante pour l'ajouter au chemin.

```
$Env:PATH += ";<path to GnuPG executable files>"
```

3. Créez un fichier texte brut local.

macOS

Sur le terminal, saisissez :

```
touch <public_key_filename.txt>
```

Ouvrez le fichier avec TextEdit.

Linux

Créez un fichier texte dans un éditeur de texte, comme gedit. Enregistrer sous `public_key_filename.txt`

Windows

Créez un fichier texte dans un éditeur de texte, comme Notepad. Enregistrer sous `public_key_filename.txt`

4. Ajoutez les éléments de clé publique PGP Amazon ECS suivants et enregistrez le fichier.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2

mQINBFq1SasBEADliGcT1NVJ1ydfN8DqebYYe9ne3dt6jqKFmKowLmm6LLGJe7HU
jGtqhCWRDkN+qPpHqdArRgDZAtn2pXY5fEipHgar4CP8QgRnRMO2fl74lmavr4Vg
7K/KH8VHlq2uRw32/B94XLEgRbGTMdWFdKuxoPCttBQaMj3LGn6Pe+6xVWRkChQu
BoQAhjBQ+bEm0kNy0LjNgjNlnL3UMAG56t8E3LANIgGgEnpNsB1UwfWluPoGZoTx
N+6pHBJrKIL/1v/ETU4FXpYw2zvhWNahxeNRnoYj3uycHkeliCrw4kj0+skizBgO
2K7oVX8Oc3j5+ZilhL/qDLXmUCb2az5cMM1mOoF8EKX5HaNuq1KfwJxqXE6NNIcO
lFTrT7QwD5fMNld3FanLgv/ZnIrsSaqJOL6zRSq8O4LN1OWBVbndExk2Kr+5kFxn
5lBPgfPgRj5hQ+KTHMa9Y8Z7yUc64BJiN6F9Nl7FJuSsfqbdkvRLsQRbcBG9qxX3
rJAEhieJzVMEUNl+EgeCkxj5xuSkNU7zw2c3hQZqEcrADLV+hvFJktOz9Gm6xzbq
lTnWWCz4xrIWtuEBA2qE+MlDheVd78a3gIsEaSTfQq0osYXaQbvlnSWOoc1y/5Zb
zizHTJIhLtUyls9WisP2s0emeHZicVMfW61EgPrJAiupgc7kyZvFt4YwfwARAQAB
tCRBbWF6b24gRUNTIDxlY3Mtc2VjdXJpdHlAYW1hem9uLmNvbT6JAhwEEAECAAYF
AlrjL0YACgkQHivRXs0TaQrg1g/+JppwPqHnlVPmv7lessB8I5UqZeD6p6uVpHd7
Bs3pcPp8BV7BdRbs3sPLt5bV1+rkqOlw+0gZ4Q/ue/YbWtOAt4qY0OcEo0HgcnaX
lsB827QIfZIVtGWMhuh94xzm/SJkvngml6KB3YJNnWP61A9qJ37/VbVVLzvcmazA
McWB4HUMNrhd0JgBCo0gIpqCbpJEvUc02Bjn23eEJsS9kC7OUAHyQkVnx4d9UzXF
4OoISF6hmQKIBoLnRrAlj5Qvs3GhvHQ0ThYq0Grk/KMJJX2CSqt7tWJ8gk1n3H3Y
SReRXJRnv7DsDDBwFgT6r5Q2HW1TBUvaoZy5hF6maD09nHcNnvBjqADzeT8Tr/Qu
bBCLzkNSYqqkpgtwv7seoD2P4n1giRvDAOEfMZpVkUr+C252IaH1HZFEz+TvBVQM
Y8OWWxmIJW+J6evjo3N1eO19UHv71jvoF8zljbI4bsL2c+QTJmOv7nRqzDQgCWyp
Id/v2dUVVTk1j9omuLBBwNJzQCB+72LcIzJhYmaP1HC4LcKQG+/f41exuItenatK
lEJQhYtyVXcBlh6Yn/wzNg2NWOwb3vqY/F7m6u9ixAwgtIMgPCDE4aJ86zrrXYFz
N2HqkTSQh77Z8KPKmyGopsmN/reMuilPdINb249nA0dzoN+nj+tTFOYCIaLaFyjs
Z0r1QAOJAjkEEwECACMFAlq1SasCGwMHCwkIBwMCAQYVCAIJCgsEFgIDAQIeAQIX
gAAKCRC86dmkLVF4T9iFEACEnkm1dNXsWUx34R3c0vamHrPxvfkyI1FlEUen8D1h
uX9xy6jCEROHWEp0rjGK4QDPgM93sWJ+s1UAKg214QRVzft0y9/DdR+twApA0fzy
uavIthGd6+03jAAo6udYDE+cZC3P7XBbDiYEWk4XAF9I1JjB8hTZUgvXBL046JhG
eM17+crgUyQeetkiOQemLbsbXQ40Bd9V7zf7XJraFd8VrwNUwNb+9KFtgAsc9rk+
YIT/PEf+YOPysgcxI4sTWghtyCulVnuGoskgDv4v73PALU0ieUrvvQVqWMRvhVx1
0X90J7cC1KOyhlEQQ1aFTgmQjmXexVTwIBm8LvysFK6YXM41KjOrlz3+6xBIm/qe
bFyLUnf4WoiuOplAaJhK9pRY+XEnGNxdtN4D26Kd0F+PLkm3Tr3Hy3b1Ok34FlGr
KVHUq1TZD7cvMnnNKEELTUcKX+1mV3an16nmAg/my1JSUt6BNK2rJpY1s/kkSGSE
XQ4zuF2IGCpvBFhYAlt5Un5zwqkwwQR3/n2kwAoDzonJcehDw/C/cGos5D0aIU7I
K2X2aTD3+pA7Mx3IMe2hqmYqRt9X42yF1PIEVRneBRJ3HDezAgJrNh0GQWRQkhIx
gz6/cTR+ekr5TptVszS9few2GpI5bCgBKBisZIssT89aw7mAKWut0Gcm4qM9/yK6
1bkCDQRatUmrARAAxNPvVwreJ2yAiFcUpdRlVhsuOgnxvs1QgsIw3H7+Pacr9Hpe
8uftYZqdC82KeSKhpHq7c8gMTMucIINtH25x9BCc73E33EjCL9Lqov1TL7+QkgHe
T+JIhZwdD8Mx2K+LVVVu/aWkNrfMuNwyDUciSI4D5QHa8T+F8fgN4OTpwYjirzel
5yoICMr9hVcbzDNv/ozKCxjx+XKgnFc3wrnDfJfntfDAT7ecwbUTL+viQKJ646s+
psiqXRYtVvYInEhLVrJ0aV6zHFoigE/Bils6/g7ru1Q6CEHqEw++APs5CcE8VzJu
```

```
WAGSVHZgun5Y9N4quR/M9Vm+IPMhTxrAg7rOvyRN9cAXfeSMf77I+XTifigNna8x
t/MOdjXr1fjF4pThEi5u6WsuRdFwjY2azEv3vevodTi4HoJReH6dFRa6y8c+UDgl
2iHiOKIpQqLbHEfQmHcDd2fix+AaJKMnPGNku9qCFEMbgSRJpXz6BfwnY1QuKE+I
R6jA0frUNt2jhiGG/F8RceXzohaaC/Cx7LUCUFWc0n7z32C9/Dtj7I1PMOacdZzz
bjJzRKO/ZDv+UN/c9dwAkllzAyPMwGBkUaY68EBstnIliW34aWm6IiHhxioVPKSp
VJfyiXPO0EXqujtHLAeChfjcns3I12YshT1dv2PafG53fp33ZdzeUgsBo+EAEQEA
AYkCHwQYAQIACQUCWrVJqwIbDAAKCRC86dmkLVF4T+ZdD/9x/8APzgNJF3o3STrF
jvnV1ycyhWYGAeBJiu7wjsNWwzMFOv15tLjB7AqeVxZn+WKDD/mIOQ45OZvnYZuy
X7DR0JszaH9wrYTxZLVruAu+t6UL0y/XQ4L1GZ9QR6+r+7t1Mvbfy7BlHbvX/gYt
Rwe/uwdibI0CagEzyX+2D3kTOlHO5XThbXaNf8AN8zha91Jt2Q2UR2X5T6JcwtMz
FBvZnl3LSmZyE0EQehS2iUurU4uWOpGppuqVnbi0jbCvCHKgDGrqZ0smKNAQng54
F365W3g8AfY48s8XQwzmcliowYX9bT8PZiEi0J4QmQh0aXkpqZyFefuWeOL2R94S
XKzr+gRh3BAULoqF+qK+IUMxTip9KTPNvYDpiC66yBiT6gFDji5Ca9pGpJXrC3xe
TXiKQ8DBWDhBPVPrruLIaenTtZEOsPc4I85yt5U9RoPTStcOr34s3w5yEaJagt6S
Gc5r9ysjkfH6+6rbi1ujxMgROSqtqr+RyB+V9A5/OgtNZc8llK6u4UoOCde8jUUW
vqWKvjJB/Kz3u4zaeNu2ZyyHaOqOuH+TETcW+jsY9IhbEzqN5yQYGi4pVmDkY5vu
lXbJnbqPKpRXgM9BecV9AMbPgbDq/5LnHJJXg+G8YQOgp4lR/hC1TEFdIp5wM8AK
CWsENyt2o1rjgMXiZOMF8A5oBLkCDQRatUuSARAAr77kj7j2QR2SZeOSlFBvV7oS
mFeSNnz9xZssqrsm6bTwSHM6YLDwc7Sdf2esDdyzONETwqrVCg+FxgL8hmo9hS4c
rR6tmrP0mOmptr+xLLsKcaP7ogIXsyZnrEAEsvW8PnfayoiPCdc3cMCR/lTnHFGA
7EuR/XLBmi7Qg9tByVYQ5Yj5wB9V4B2yeCt3XtzPqeLKvaxl7PNelaHGJQY/xo+m
V0bndxf9IY+4oFJ4blD32WqvyxESo7vW6WBh7oqv3Zbm0yQrr8a6mDBpqLkvWwNI
3kpJR974tg5o5LfDu1BeeyHWPSGm4U/G4JB+JIG1ADy+RmoWEt4BqTCZ/knnoGvw
D5sTCxbKdmuOmhGyTssoG+3OOcGYHV7pWYPhazKHMPm201xKCjH1RfzRULzGKjD+
yMLT1I3AXFmLmZJXikAOlvE3/wgMqCXscbycbLjLD/bXIuFWo3rzoezeXjgi/DJx
jKBAyBTYO5nMcth1O9oaFd9d0HbsOUDkIMnsgGBE766Piro6MHo0T0rXl07Tp4pI
rwuSOsc6XzCzdImj0Wc6axS/HeUKRXWdXJwno5awTwXKRJMXGfhCvSvbcbc2Wx+L
IKvmB7EB4K3fmjFFE67yolmiw2qRcUBfygtH3eL5XZU28MiCpue8Y8GKJoBAUyvf
KeM1rO8Jm3iRAc5a/D0AEQEAAYkEPgQYAQIACQUCWrVLkgIbAgIpCRC86dmkLVF4
T8FdIAQZAQIABgUCWrVLkgAKCRDePL1hra+LjtHYD/9MucxdFe6bXO1dQR4tKhhQ
P0LRqy6zlBY9ILCLowNdGZdqorogUiUymgn3VhEhVtxTOoHcN7qOuM01PNsRnOeS
EYjf8Xrb1clzkD6xULwmOclTb9bBxnBc/4PFvHAbZW3QzusaZniNgkuxt6BTfloS
Of4inq71kjmGK+TlzQ6mUMQUg228NUQC+a84EPqYyAeY1sgvgB7hJBhYL0QAxhcW
6m20Rd8iEc6HyzJ3yCOCsKip/nRWAbf0OvfHfRBp0+m0ZwnJM8cPRFjOqqzFpKH9
HpDmTrC4wKP1+TL52LyEqNh4yZitXmZNV7giSRIkk0eDSko+bFy6VbMzKUMkUJK3
D3eHFAMkujmbfJmSMTJOPGn5SB1HyjCZNx6bhIIbQyEUB9gKCmUFaqXKwKpF6rj0
iQXAJxLR/shZ5Rk96VxzOphUl7T90m/PnUEEPwq8KsBhnMRgxa0RFidDP+n9fgtv
HLmrOqX9zBCVXh0mdWYLrWvmzQFWzG7AoE55fkf8nAEPsalrCdtaNUBHRXA0OQxG
AHMOdJQQvBsmqMvuAdjkDWpFu5y0My5ddU+hiUzUyQLjL5Hhd5LOUDdewlZgIw1j
xrEAUzDKetnemM8GkHxDgg8koev5frmShJuce7vSjKpCNg3EIJSgqMOPFjJuLWtZ
vjHeDNbJy6uNL65ckJy6WhGjEADS2WAW1D6Tfekkc21SsIXk/LqEpLMR/0g5OUif
wcEN1rS9IJXBwIy8MelN9qr5KcKQLmfdfBNEyyceBhyVl0MDyHOKC+7PofMtkGBq
13QieRHv5GJ8LB3fclqHV8pwTTo3Bc8z2g0TjmUYAN/ixETdReDoKavWJYSE9yoM
aaJu279ioVTrwpECse0XkiRyKToTjwOb73CGkBZZpJyqux/rmCV/fp4ALdSW8zbz
```

```
FJVORaivhoWwzjpfQKhwcU9lABXi2UvVm14v0AfeI7oiJPSU1zM4fEny4oiIBXlR
zhFNih1UjIu82X16mTm3BwbIga/s1fnQRGzyhqUIMii+mWra23EwjChaxpvjjcUH
5ilLc5Zq781aCYRygYQw+hu5nFkOH1R+Z50Ubxjd/aqUfnGIAX7kPMD3Lof4KldD
Q8ppQriUvxVo+4nPV6rpTy/PyqCLWDjkguHpJsEFsMkwajrAz0QNSAU5CJ0G2Zu4
yxvYlumHCEl7nbFrm0vIiA75Sa8KnywTDsyZsu3XcOcf3g+g1xWTpjJqy2bYXlqz
9uDOWtArWHOis6bq8l9RE6xr1RBVXS6uqgQIZFBGyq66b0dIq4D2JdsUvgEMaHbc
e7tBfeB1CMBdA64e9Rq7bFR7Tvt8gasCZYlNr3lydh+dFHIEkH53HzQe6l88HEic
+0jVnLkCDQRa55wJARAAyLya2Lx6gyoWoJN1a6740q3o8e9d4KggQOfGMTCflmeq
ivuzgN+3DZHN+9ty2KxXMtn0mhHBerZdbNJyjMNT1gAgrhPNB4HtXBXum2wS57WK
DNmade914L7FWTPAWBG2Wn448OEHTqsClICXXWy9IICgclAEyIq0Yq5mAdTEgRJS
Z8t4GpwtDL9gNQyFXaWQmDmkAsCygQMvhAlmu9xOIzQG5CxSnZFk7zcuL60k14Z3
Cmt49k4T/7ZU8goWi8tt+rU78/IL3J/fF9+1civ1OwuUidgfPCSvOUW1JojsdCQA
L+RZJcoXq7lfOFj/eNjeOSstCTDPfTCL+kThE6E5neDtbQHBYkEX1BRiTedsV4+M
ucgiTrdQFWKf89G72xdv8ut9AYYQ2BbEYU+JAYhUH8rYYui2dHKJIgjNvJscuUWb
+QEqJIRleJRhrO+/CHgMs4fZAkWF1VFhKBkcKmEjLn1f7EJJUUW84ZhKXjO/AUPX
1CHsNjziRceuJCJYox1cwsoq6jTE50GiNzcIxTn9xUc0UMKFeggNAFys1K+TDTm3
Bzo8H5ucjCUEmUm9lhkGwqTZgOlRX5eqPX+JBoSaObqhgqCa5IPinKRa6MgoFPHK
6sYKqroYwBGgZm6Js5chpNchvJMs/3WXNOEVg0J3z3vP0DMhxqWm+r+n9zlW8qsA
EQEAAYkEPgQYAQgACQUCWuecCQIbAgIpCRC86dmkLVF4T8FdIAQZAQgABgUCWuec
CQAKCRBQ3szEcQ5hr+ykD/4tOLRHFHXuKUcxgGaubUcVtsFrwBKma1cYjqaPms8u
6Sk0wfGRI32G/GhOrp0Ts/MOkbObq6VLTh8N5Yc/53MEl8zQFw9Y5AmRoW4PZXER
ujs5s7p4oR7xHMihMjCCBn1bvrR+34YPfgzTcgLiOEFHYT8UTxwnGmXOvNkMM7md
xD3CV5q6VAte8WKBo/220II3fcQlc9r/oWX4kXXkb0v9hoGwKbDJ1tzqTPrp/xFt
yohqnvImpnlz+Q9zXmbrWYL9/g8VCmW/NN2gju2G3Lu/TlFUWIT4v/5OPK6TdeNb
VKJO4+S8bTayqSG9CML1S57KSgCo5HUhQWeSNHI+fpe5oX6FALPT9JLDce8OZz1i
cZZ0MELP37mOOQun0AlmHm/hVzf0f311PtbzcqWaE51tJvgUR/nZFo6Ta3O5Ezhs
3VlEJNQ1Ijf/6DH87SxvAoRIARCuZd0qxBcDK0avpFzUtbJd24lRA3WJpkEiMqKv
RDVZkE4b6TW61f0o+LaVfK6E8oLpixegS4fiqC16mFrOdyRk+RJJfIUyz0WTDVmt
g0U1CO1ezokMSqkJ7724pyjr2xf/r9/sC6aOJwB/lKgZkJfC6NqL7TlxVA31dUga
LEOvEJTTE4gl+tYtfsCDvALCtqL0jduSkUo+RXcBItmXhA+tShW0pbS2Rtx/ixua
KohVD/0R4QxiSwQmICNtm9mw9ydIl1yjYXX5a9x4wMJracNY/LBybJPFnZnT4dYR
z4XjqysDwvvYZByaWoIe3QxjX84V6MlI2IdAT/xImu8gbaCI8tmyfpIrLnPKiR9D
VFYfGBXuAX7+HgPPSFtrHQONCALxxzlbNpS+zxt9r0MiLgcLyspWxSdmoYGZ6nQP
RO5Nm/ZVS+u2imPCRzNUZEMa+dlE6kHx0rS0dPiuJ4O7NtPeYDKkoQtNagspsDvh
cK7CSqAiKMq06UBTxqlTSRkm62eOCtcs3p3OeHu5GRZF1uzTET0ZxYkaPgdrQknx
ozjP5mC7X+45lcCfmcVt94TFNL5HwEUVJpmOgmzILCI8yoDTWzloo+i+fPFsXX4f
kynhE83mSEcr5VHFYrTY3mQXGmNJ3bCLuc/jq7ysGq69xiKmTlUeXFm+aojcRO5i
zyShIRJZ0GZfuzDYFDbMV9amA/YQGygLw//zP5ju5SW26dNxlf3MdFQE5JJ86rn9
MgZ4gcpazHEVUsbZsgkLizRp9imUiH8ymLqAXnfRGlU/LpNSefnvDFTtEIRcpOHc
bhayG0bk51Bd4mioOXnIsKy4j63nJXA27x5EVVHQ1sYRN8Ny4Fdr2tMAmj2O+X+J
qX2yy/UX5nSPU492e2CdZ1UhoU0SRFY3bxKHKB7SDbVeav+K5g==
=Gi5D
-----END PGP PUBLIC KEY BLOCK-----
```

Détails de la clé publique PGP Amazon ECS pour référence :

```
Key ID: BCE9D9A42D51784F
Type: RSA
Size: 4096/4096
Expires: Never
User ID: Amazon ECS
Key fingerprint: F34C 3DDA E729 26B0 79BE AEC6 BCE9 D9A4 2D51 784F
```

5. Importez le fichier avec la clé publique PGP Amazon ECS avec la commande suivante dans le terminal.

```
gpg --import <public_key_filename.txt>
```

6. Téléchargez les signatures de la AWS CLI Copilot. Les signatures sont des signatures PGP détachées ASCII, stockées dans des fichiers portant l'extension `.asc`. Le fichier de signatures a le même nom que son ficher exécutable correspondant, avec l'extension `.asc`.

   macOS

   Pour les systèmes macOS, exécutez la commande suivante :

   ```
   sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/
   download/copilot-darwin.asc
   ```

   Linux

   Pour les systèmes Linux x86 (64 bits), exécutez la commande suivante :

   ```
   sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/
   download/copilot-linux.asc
   ```

   Pour les systèmes ARM Linux, exécutez la commande suivante :

   ```
   sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/
   download/copilot-linux-arm64.asc
   ```

   Windows

   Exécutez la commande suivante à l'aide de PowerShell.

```
Invoke-WebRequest -OutFile 'C:\copilot\copilot.asc' https://github.com/aws/
copilot-cli/releases/latest/download/copilot-windows.exe.asc
```

7. Vérifiez la signature avec la commande suivante :

   - Pour les systèmes Mac OS et Linux :

     ```
     gpg --verify copilot.asc /usr/local/bin/copilot
     ```

   - Pour les systèmes Windows :

     ```
     gpg --verify 'C:\copilot\copilot.asc' 'C:\copilot\copilot.exe'
     ```

   Sortie attendue :

   ```
   gpg: Signature made Tue Apr  3 13:29:30 2018 PDT
   gpg:                using RSA key DE3CBD61ADAF8B8E
   gpg: Good signature from "Amazon ECS <ecs-security@amazon.com>" [unknown]
   gpg: WARNING: This key is not certified with a trusted signature!
   gpg:          There is no indication that the signature belongs to the owner.
   Primary key fingerprint: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
        Subkey fingerprint: EB3D F841 E2C9 212A 2BD4  2232 DE3C BD61 ADAF 8B8E
   ```

   > **Important**
   >
   > L'avertissement présent dans la sortie est prévu et ne constitue pas un problème. Cet avertissement est déclenché, car il n'y a pas de chaîne de confiance entre votre clé PGP personnelle (si vous en avez une) et la clé PGP Amazon ECS. Pour de plus amples informations, consultez Web of trust.

8. Pour les installations Windows, exécutez la commande suivante sur Powershell pour ajouter le répertoire AWS Copilot au chemin.

   ```
   $Env:PATH += ";<path to Copilot executable files>"
   ```

Déploiement d'un exemple d'application Amazon ECS à l'aide de la CLI AWS Copilot

Après avoir installé la AWS CLI Copilot, vous pouvez suivre ces étapes pour déployer un exemple d'application, vérifier le déploiement et nettoyer les ressources.

Prérequis

Avant de commencer, assurez-vous de remplir les prérequis suivants :

- Installation et configuration de l' AWS CLI. Pour plus d'informations, consultez Interface de ligne de commande AWS.
- Exécutez `aws configure` pour configurer un profil par défaut que la AWS CLI Copilot utilisera pour gérer votre application et vos services.
- Installez et exécutez Docker. Pour plus d'informations, veuillez consulter Get started with Docker.

Déployez un exemple d'application Amazon ECS à l'aide d'une seule commande

1. Déployez un exemple d'application Web cloné à partir d'un GitHub référentiel à l'aide de la commande suivante. Pour plus d'informations sur AWS Copilot `init` et ses drapeaux, consultez la documentation AWS Copilot.

```
git clone https://github.com/aws-samples/aws-copilot-sample-service.git demo-app &&
 \
cd demo-app &&                                \
copilot init --app demo                       \
  --name api                                  \
  --type 'Load Balanced Web Service'          \
  --dockerfile './Dockerfile'                 \
  --port 80                                   \
  --tag  latest                               \
  --deploy
```

2. Une fois le déploiement terminé, la AWS CLI Copilot renvoie une URL que vous pouvez utiliser pour vérifier le déploiement. Vous pouvez également utiliser les commandes suivantes pour vérifier l'état de l'application.

   - Répertoriez toutes vos applications AWS Copilot.

```
copilot app ls
```

- Affichez des informations sur les environnements et les services dans votre application.

```
copilot app show
```

- Affichez des informations sur vos environnements.

```
copilot env ls
```

- Affiche des informations sur le service, y compris les points de terminaison, la capacité et les ressources connexes.

```
copilot svc show
```

- Liste de tous les services d'une application.

```
copilot svc ls
```

- Affichez les journaux d'un service déployé.

```
copilot svc logs
```

- Affichez l'état du service.

```
copilot svc status
```

3. Lorsque vous aurez terminé cette démonstration, exécutez la commande suivante pour nettoyer les ressources associées et éviter de payer des frais pour les ressources non utilisées.

```
copilot app delete
```

Utilisation d'Amazon ECS avec AWS CloudFormation

Amazon ECS est intégré à AWS CloudFormation un service que vous pouvez utiliser pour modéliser et configurer des AWS ressources à l'aide de modèles que vous définissez. AWS CloudFormation utilise des modèles sous forme de fichier texte `YAML` ou de fichier texte `JSON` formaté. Les modèles sont comme des plans pour la AWS ressource que vous souhaitez créer. Lorsque vous créez et soumettez un modèle, il AWS CloudFormation crée une pile. Vous gérez les ressources que vous avez définies dans votre modèle par le biais de la pile. Lorsque vous souhaitez créer, mettre à jour ou supprimer une ressource, vous créez, mettez à jour ou supprimez la pile créée à partir de cette ressource. Lorsqu'il s'agit de mettre à jour vos piles, vous devez d'abord créer un ensemble de modifications. Les ensembles de modifications vous indiquent ce qui est impacté par la modification avant que vous ne l'apportiez. Cela vous empêche de supprimer des bases de données accidentellement en modifiant le nom de votre base de données, par exemple. Pour plus d'informations sur les modèles, les piles et les ensembles de modifications, consultez la section AWS CloudFormation Fonctionnement du guide de l'AWS CloudFormation utilisateur.

En l'utilisant AWS CloudFormation, vous pouvez passer moins de temps à créer et à gérer vos ressources et votre infrastructure. Vous pouvez créer un modèle qui décrit toutes les AWS ressources souhaitées, telles que les clusters Amazon ECS, les définitions de tâches et les services. AWS CloudFormation Il s'occupe ensuite du provisionnement et de la configuration de ces ressources pour vous.

AWS CloudFormation vous permet également de réutiliser votre modèle pour configurer vos ressources Amazon ECS de manière cohérente et reproductible. Vous décrivez vos ressources une seule fois, puis vous réapprovisionnez les mêmes ressources sur plusieurs Comptes AWS et Régions AWS.

AWS CloudFormation les modèles peuvent être utilisés à la fois avec AWS Management Console ou AWS Command Line Interface pour créer des ressources.

Pour en savoir plus AWS CloudFormation, consultez les ressources suivantes :

- AWS CloudFormation
- AWS CloudFormation Guide de l'utilisateur
- AWS CloudFormation Guide de l'utilisateur de l'interface de ligne de commande

Rubriques

- AWS CloudFormation exemples de modèles pour Amazon ECS
- Création de ressources Amazon ECS à l'aide de la AWS CloudFormation console
- Création de ressources Amazon ECS à l'aide de AWS CLI commandes pour AWS CloudFormation

AWS CloudFormation exemples de modèles pour Amazon ECS

Pour fournir et configurer des ressources pour Amazon ECS et les services associés, assurez-vous de connaître les AWS CloudFormation modèles. Pour plus d'informations, consultez les AWS CloudFormation modèles dans le guide de AWS CloudFormation l'utilisateur. AWS CloudFormation les modèles sont des fichiers texte au format JSON ou YAML qui décrivent les ressources que vous souhaitez fournir dans vos AWS CloudFormation piles. Si vous ne connaissez pas le format JSON ou YAML, ou les deux, vous pouvez commencer AWS Infrastructure Composer à utiliser des AWS CloudFormation modèles. Pour plus d'informations, voir Création visuelle de modèles avec Infrastructure Composer dans le guide de AWS CloudFormation l'utilisateur.

Vous pouvez créer des clusters Amazon ECS, des définitions de tâches et des services à l'aide de AWS CloudFormation. Les rubriques suivantes incluent des modèles qui montrent comment créer des ressources avec différentes configurations. Vous pouvez créer ces ressources à l'aide de ces modèles à l'aide de la AWS CloudFormation console ou du AWS CLI. Pour plus d'informations sur la création de ressources à l'aide de la AWS CloudFormation console, consultez le guide de AWS CloudFormation l'utilisateur.

Les rubriques suivantes répertorient des exemples de modèles pour les définitions de tâches, les clusters et les services Amazon ECS.

Rubriques

- Définitions de tâche
- Clusters
- Services
- Rôles IAM pour Amazon ECS

Définitions de tâche

Une définition de tâche est un plan pour votre application qui décrit les paramètres et un ou plusieurs conteneurs qui constituent votre application. Vous trouverez ci-dessous des exemples AWS

CloudFormation de modèles pour les définitions de tâches Amazon ECS. Pour plus d'informations sur les définitions de tâches Amazon ECS, consultez Définitions de tâche Amazon ECS.

Définition de la tâche Fargate Linux

Vous pouvez utiliser le modèle suivant pour créer un exemple de tâche Fargate Linux.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "ECS Task Definition with parameterized values",
  "Parameters": {
    "ContainerImage": {
      "Type": "String",
      "Default": "public.ecr.aws/docker/library/httpd:2.4",
      "Description": "The container image to use for the task"
    },
    "ContainerCpu": {
      "Type": "Number",
      "Default": 256,
      "Description": "The number of CPU units to reserve for the container",
      "AllowedValues": [256, 512, 1024, 2048, 4096]
    },
    "ContainerMemory": {
      "Type": "Number",
      "Default": 512,
      "Description": "The amount of memory (in MiB) to reserve for the container",
      "AllowedValues": [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192]
    },
    "TaskFamily": {
      "Type": "String",
      "Default": "task-definition-cfn",
      "Description": "The name of the task definition family"
    },
    "ContainerName": {
      "Type": "String",
      "Default": "sample-fargate-app",
      "Description": "The name of the container"
    },
    "ContainerPort": {
      "Type": "Number",
      "Default": 80,
```

```
      "Description": "The port number on the container"
    },
    "HostPort": {
      "Type": "Number",
      "Default": 80,
      "Description": "The port number on the host"
    },
    "ExecutionRoleArn": {
      "Type": "String",
      "Default": "arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole",
      "Description": "The ARN of the task execution role"
    },
    "LogGroup": {
      "Type": "String",
      "Default": "/ecs/fargate-task-definition",
      "Description": "The CloudWatch log group for container logs"
    },
    "NetworkMode": {
      "Type": "String",
      "Default": "awsvpc",
      "Description": "The Docker networking mode to use",
      "AllowedValues": ["awsvpc", "bridge", "host", "none"]
    },
    "OperatingSystemFamily": {
      "Type": "String",
      "Default": "LINUX",
      "Description": "The operating system for the task",
      "AllowedValues": ["LINUX", "WINDOWS_SERVER_2019_FULL",
 "WINDOWS_SERVER_2019_CORE", "WINDOWS_SERVER_2022_FULL", "WINDOWS_SERVER_2022_CORE"]
    }
  },
  "Resources": {
    "ECSTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ContainerDefinitions": [
          {
            "Command": [
              "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</
title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && -
foreground\""
```

```
            ],
            "EntryPoint": [
              "sh",
              "-c"
            ],
            "Essential": true,
            "Image": {"Ref": "ContainerImage"},
            "LogConfiguration": {
              "LogDriver": "awslogs",
              "Options": {
                "mode": "non-blocking",
                "max-buffer-size": "25m",
                "awslogs-create-group": "true",
                "awslogs-group": {"Ref": "LogGroup"},
                "awslogs-region": {"Ref": "AWS::Region"},
                "awslogs-stream-prefix": "ecs"
              }
            },
            "Name": {"Ref": "ContainerName"},
            "PortMappings": [
              {
                "ContainerPort": {"Ref": "ContainerPort"},
                "HostPort": {"Ref": "HostPort"},
                "Protocol": "tcp"
              }
            ]
          }
        ],
        "Cpu": {"Ref": "ContainerCpu"},
        "ExecutionRoleArn": {"Ref": "ExecutionRoleArn"},
        "Family": {"Ref": "TaskFamily"},
        "Memory": {"Ref": "ContainerMemory"},
        "NetworkMode": {"Ref": "NetworkMode"},
        "RequiresCompatibilities": [
          "FARGATE"
        ],
        "RuntimePlatform": {
          "OperatingSystemFamily": {"Ref": "OperatingSystemFamily"}
        }
      }
    }
  },
  "Outputs": {
    "TaskDefinitionArn": {
```

```
      "Description": "The ARN of the created task definition",
      "Value": {"Ref": "ECSTaskDefinition"}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: 'ECS Task Definition to deploy a sample app'
Parameters:
  ContainerImage:
    Type: String
    Default: 'public.ecr.aws/docker/library/httpd:2.4'
    Description: The container image to use for the task
  ContainerCpu:
    Type: Number
    Default: 256
    Description: The number of CPU units to reserve for the container
    AllowedValues: [256, 512, 1024, 2048, 4096]
  ContainerMemory:
    Type: Number
    Default: 512
    Description: The amount of memory (in MiB) to reserve for the container
    AllowedValues: [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192]
  TaskFamily:
    Type: String
    Default: 'task-definition-cfn'
    Description: The name of the task definition family
  ContainerName:
    Type: String
    Default: 'sample-fargate-app'
    Description: The name of the container
  ContainerPort:
    Type: Number
    Default: 80
    Description: The port number on the container
  HostPort:
    Type: Number
    Default: 80
    Description: The port number on the host
  ExecutionRoleArn:
    Type: String
```

```
    Default: 'arn:aws:iam::111122223333:role/ecsTaskExecutionRole'
    Description: The ARN of the task execution role
  LogGroup:
    Type: String
    Default: '/ecs/fargate-task-definition'
    Description: The CloudWatch log group for container logs
  NetworkMode:
    Type: String
    Default: 'awsvpc'
    Description: The Docker networking mode to use
    AllowedValues: ['awsvpc', 'bridge', 'host', 'none']
  OperatingSystemFamily:
    Type: String
    Default: 'LINUX'
    Description: The operating system for the task
    AllowedValues: ['LINUX', 'WINDOWS_SERVER_2019_FULL', 'WINDOWS_SERVER_2019_CORE',
 'WINDOWS_SERVER_2022_FULL', 'WINDOWS_SERVER_2022_CORE']
Resources:
  ECSTaskDefinition:
    Type: 'AWS::ECS::TaskDefinition'
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
```

```
              awslogs-stream-prefix: ecs
          Name: !Ref ContainerName
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              HostPort: !Ref HostPort
              Protocol: tcp
      Cpu: !Ref ContainerCpu
      ExecutionRoleArn: !Ref ExecutionRoleArn
      Family: !Ref TaskFamily
      Memory: !Ref ContainerMemory
      NetworkMode: !Ref NetworkMode
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: !Ref OperatingSystemFamily
Outputs:
  TaskDefinitionArn:
    Description: The ARN of the created task definition
    Value: !Ref ECSTaskDefinition
```

Définition des tâches Amazon EFS

Vous pouvez utiliser le modèle suivant pour créer une tâche qui utilise un système de fichiers Amazon EFS que vous avez créé. Pour plus d'informations sur l'utilisation des volumes Amazon EFS avec Amazon ECS, consultez Utiliser les volumes Amazon EFS avec Amazon ECS.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Create a task definition for a web server with parameterized
 values.",
  "Parameters": {
    "ExecutionRoleArn": {
      "Type": "String",
      "Default": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
      "Description": "The ARN of the task execution role"
    },
    "NetworkMode": {
      "Type": "String",
      "Default": "awsvpc",
      "Description": "The Docker networking mode to use",
```

```
      "AllowedValues": ["awsvpc", "bridge", "host", "none"]
    },
    "TaskFamily": {
      "Type": "String",
      "Default": "my-ecs-task",
      "Description": "The name of the task definition family"
    },
    "ContainerCpu": {
      "Type": "String",
      "Default": "256",
      "Description": "The number of CPU units to reserve for the container",
      "AllowedValues": ["256", "512", "1024", "2048", "4096"]
    },
    "ContainerMemory": {
      "Type": "String",
      "Default": "512",
      "Description": "The amount of memory (in MiB) to reserve for the container",
      "AllowedValues": ["512", "1024", "2048", "3072", "4096", "5120", "6144",
 "7168", "8192"]
    },
    "ContainerName": {
      "Type": "String",
      "Default": "nginx",
      "Description": "The name of the container"
    },
    "ContainerImage": {
      "Type": "String",
      "Default": "public.ecr.aws/nginx/nginx:latest",
      "Description": "The container image to use for the task"
    },
    "ContainerPort": {
      "Type": "Number",
      "Default": 80,
      "Description": "The port number on the container"
    },
    "InitProcessEnabled": {
      "Type": "String",
      "Default": "true",
      "Description": "Whether to enable the init process inside the container",
      "AllowedValues": ["true", "false"]
    },
    "EfsVolumeName": {
      "Type": "String",
      "Default": "efs-volume",
```

```
      "Description": "The name of the EFS volume"
    },
    "EfsContainerPath": {
      "Type": "String",
      "Default": "/usr/share/nginx/html",
      "Description": "The path in the container where the EFS volume will be
mounted"
    },
    "LogGroup": {
      "Type": "String",
      "Default": "LogGroup",
      "Description": "The CloudWatch log group for container logs"
    },
    "LogStreamPrefix": {
      "Type": "String",
      "Default": "efs-task",
      "Description": "The prefix for the log stream"
    },
    "EfsFilesystemId": {
      "Type": "String",
      "Default": "fs-1234567890abcdef0",
      "Description": "The ID of the EFS filesystem"
    },
    "EfsRootDirectory": {
      "Type": "String",
      "Default": "/",
      "Description": "The root directory in the EFS filesystem"
    },
    "EfsTransitEncryption": {
      "Type": "String",
      "Default": "ENABLED",
      "Description": "Whether to enable transit encryption for EFS",
      "AllowedValues": ["ENABLED", "DISABLED"]
    }
  },
  "Resources": {
    "ECSTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ExecutionRoleArn": {"Ref": "ExecutionRoleArn"},
        "NetworkMode": {"Ref": "NetworkMode"},
        "RequiresCompatibilities": ["FARGATE"],
        "Family": {"Ref": "TaskFamily"},
        "Cpu": {"Ref": "ContainerCpu"},
```

```
        "Memory": {"Ref": "ContainerMemory"},
        "ContainerDefinitions": [
          {
            "Name": {"Ref": "ContainerName"},
            "Image": {"Ref": "ContainerImage"},
            "Essential": true,
            "PortMappings": [
              {
                "ContainerPort": {"Ref": "ContainerPort"},
                "Protocol": "tcp"
              }
            ],
            "LinuxParameters": {
              "InitProcessEnabled": {"Ref": "InitProcessEnabled"}
            },
            "MountPoints": [
              {
                "SourceVolume": {"Ref": "EfsVolumeName"},
                "ContainerPath": {"Ref": "EfsContainerPath"}
              }
            ],
            "LogConfiguration": {
              "LogDriver": "awslogs",
              "Options": {
                "mode": "non-blocking",
                "max-buffer-size": "25m",
                "awslogs-group": {"Ref": "LogGroup"},
                "awslogs-region": {"Ref": "AWS::Region"},
                "awslogs-create-group": "true",
                "awslogs-stream-prefix": {"Ref": "LogStreamPrefix"}
              }
            }
          }
        ],
        "Volumes": [
          {
            "Name": {"Ref": "EfsVolumeName"},
            "EFSVolumeConfiguration": {
              "FilesystemId": {"Ref": "EfsFilesystemId"},
              "RootDirectory": {"Ref": "EfsRootDirectory"},
              "TransitEncryption": {"Ref": "EfsTransitEncryption"}
            }
          }
        ]
```

```
      }
    }
  },
  "Outputs": {
    "TaskDefinitionArn": {
      "Description": "The ARN of the created task definition",
      "Value": {"Ref": "ECSTaskDefinition"}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: Create a task definition for a web server with parameterized values.
Parameters:
  ExecutionRoleArn:
    Type: String
    Default: arn:aws:iam::123456789012:role/ecsTaskExecutionRole
    Description: The ARN of the task execution role
  NetworkMode:
    Type: String
    Default: awsvpc
    Description: The Docker networking mode to use
    AllowedValues: [awsvpc, bridge, host, none]
  TaskFamily:
    Type: String
    Default: my-ecs-task
    Description: The name of the task definition family
  ContainerCpu:
    Type: String
    Default: "256"
    Description: The number of CPU units to reserve for the container
    AllowedValues: ["256", "512", "1024", "2048", "4096"]
  ContainerMemory:
    Type: String
    Default: "512"
    Description: The amount of memory (in MiB) to reserve for the container
    AllowedValues: ["512", "1024", "2048", "3072", "4096", "5120", "6144", "7168",
 "8192"]
  ContainerName:
    Type: String
    Default: nginx
```

```
    Description: The name of the container
  ContainerImage:
    Type: String
    Default: public.ecr.aws/nginx/nginx:latest
    Description: The container image to use for the task
  ContainerPort:
    Type: Number
    Default: 80
    Description: The port number on the container
  InitProcessEnabled:
    Type: String
    Default: "true"
    Description: Whether to enable the init process inside the container
    AllowedValues: ["true", "false"]
  EfsVolumeName:
    Type: String
    Default: efs-volume
    Description: The name of the EFS volume
  EfsContainerPath:
    Type: String
    Default: /usr/share/nginx/html
    Description: The path in the container where the EFS volume will be mounted
  LogGroup:
    Type: String
    Default: LogGroup
    Description: The CloudWatch log group for container logs
  LogStreamPrefix:
    Type: String
    Default: efs-task
    Description: The prefix for the log stream
  EfsFilesystemId:
    Type: String
    Default: fs-1234567890abcdef0
    Description: The ID of the EFS filesystem
  EfsRootDirectory:
    Type: String
    Default: /
    Description: The root directory in the EFS filesystem
  EfsTransitEncryption:
    Type: String
    Default: ENABLED
    Description: Whether to enable transit encryption for EFS
    AllowedValues: [ENABLED, DISABLED]
Resources:
```

```
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ExecutionRoleArn: !Ref ExecutionRoleArn
      NetworkMode: !Ref NetworkMode
      RequiresCompatibilities:
        - FARGATE
      Family: !Ref TaskFamily
      Cpu: !Ref ContainerCpu
      Memory: !Ref ContainerMemory
      ContainerDefinitions:
        - Name: !Ref ContainerName
          Image: !Ref ContainerImage
          Essential: true
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              Protocol: tcp
          LinuxParameters:
            InitProcessEnabled: !Ref InitProcessEnabled
          MountPoints:
            - SourceVolume: !Ref EfsVolumeName
              ContainerPath: !Ref EfsContainerPath
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-create-group: "true"
              awslogs-stream-prefix: !Ref LogStreamPrefix
      Volumes:
        - Name: !Ref EfsVolumeName
          EFSVolumeConfiguration:
            FilesystemId: !Ref EfsFilesystemId
            RootDirectory: !Ref EfsRootDirectory
            TransitEncryption: !Ref EfsTransitEncryption
Outputs:
  TaskDefinitionArn:
    Description: The ARN of the created task definition
    Value: !Ref ECSTaskDefinition
```

Clusters

Un cluster Amazon ECS est un regroupement logique de tâches ou de services. Vous pouvez utiliser les modèles suivants pour créer des clusters avec différentes configurations. Pour plus d'informations sur les clusters Amazon ECS, consultezClusters Amazon ECS.

Création d'un cluster vide avec les paramètres par défaut

Vous pouvez utiliser le modèle suivant pour créer un cluster vide avec les paramètres par défaut.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": "MyEmptyCluster"
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  ECSCluster:
    Type: 'AWS::ECS::Cluster'
    Properties:
      ClusterName: MyEmptyCluster
```

Créez un cluster vide grâce au chiffrement du stockage géré et à Container Insights amélioré

Vous pouvez utiliser le modèle suivant pour créer un cluster avec un stockage géré au niveau du cluster et activé Container Insights amélioré. Le chiffrement au niveau du cluster s'applique aux volumes de données gérés par Amazon ECS tels que les volumes Amazon EBS. Pour plus

d'informations sur le chiffrement Amazon EBS, consultezChiffrez les données stockées dans les volumes Amazon EBS associés aux tâches Amazon ECS. Pour plus d'informations sur l'utilisation de Container Insights avec une observabilité améliorée, consultez. Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
        "Cluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": "EncryptedEnhancedCluster",
                "ClusterSettings": [
                    {
                        "Name": "containerInsights",
                        "Value": "enhanced"
                    }
                ],
                "Configuration": {
                    "ManagedStorageConfiguration": {
                        "KmsKeyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
                    }
                }
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  Cluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: EncryptedEnhancedCluster
      ClusterSettings:
        - Name: containerInsights
          Value: enhanced
      Configuration:
        ManagedStorageConfiguration:
```

```
            KmsKeyId: a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Créez un cluster avec le AL2 023 Amazon ECS-Optimized-AMI

Vous pouvez utiliser le modèle suivant pour créer un cluster qui utilise un fournisseur de capacité qui lance AL2 023 instances sur Amazon EC2.

> **Important**
>
> Pour obtenir la dernière AMI IDs, consultez l'AMI optimisée pour Amazon ECS dans le manuel Amazon Elastic Container Service Developer Guide.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "EC2 ECS cluster that starts out empty, with no EC2 instances
 yet. An ECS capacity provider automatically launches more EC2 instances as required
 on the fly when you request ECS to launch services or standalone tasks.",
    "Parameters": {
        "InstanceType": {
            "Type": "String",
            "Description": "EC2 instance type",
            "Default": "t2.medium",
            "AllowedValues": [
                "t1.micro",
                "t2.2xlarge",
                "t2.large",
                "t2.medium",
                "t2.micro",
                "t2.nano",
                "t2.small",
                "t2.xlarge",
                "t3.2xlarge",
                "t3.large",
                "t3.medium",
                "t3.micro",
                "t3.nano",
                "t3.small",
                "t3.xlarge"
```

```
            ]
        },
        "DesiredCapacity": {
            "Type": "Number",
            "Default": "0",
            "Description": "Number of EC2 instances to launch in your ECS cluster."
        },
        "MaxSize": {
            "Type": "Number",
            "Default": "100",
            "Description": "Maximum number of EC2 instances that can be launched in
 your ECS cluster."
        },
        "ECSAMI": {
            "Description": "The Amazon Machine Image ID used for the cluster",
            "Type": "AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>",
            "Default": "/aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended/image_id"
        },
        "VpcId": {
            "Type": "AWS::EC2::VPC::Id",
            "Description": "VPC ID where the ECS cluster is launched",
            "Default": "vpc-1234567890abcdef0"
        },
        "SubnetIds": {
            "Type": "List<AWS::EC2::Subnet::Id>",
            "Description": "List of subnet IDs where the EC2 instances will be
 launched",
            "Default": "subnet-021345abcdef67890"
        }
    },
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterSettings": [
                    {
                        "Name": "containerInsights",
                        "Value": "enabled"
                    }
                ]
            }
        },
        "ECSAutoScalingGroup": {
```

```
            "Type": "AWS::AutoScaling::AutoScalingGroup",
            "DependsOn": [
                "ECSCluster",
                "EC2Role"
            ],
            "Properties": {
                "VPCZoneIdentifier": {
                    "Ref": "SubnetIds"
                },
                "LaunchTemplate": {
                    "LaunchTemplateId": {
                        "Ref": "ContainerInstances"
                    },
                    "Version": {
                        "Fn::GetAtt": [
                            "ContainerInstances",
                            "LatestVersionNumber"
                        ]
                    }
                },
                "MinSize": 0,
                "MaxSize": {
                    "Ref": "MaxSize"
                },
                "DesiredCapacity": {
                    "Ref": "DesiredCapacity"
                },
                "NewInstancesProtectedFromScaleIn": true
            },
            "UpdatePolicy": {
                "AutoScalingReplacingUpdate": {
                    "WillReplace": "true"
                }
            }
        },
        "ContainerInstances": {
            "Type": "AWS::EC2::LaunchTemplate",
            "Properties": {
                "LaunchTemplateName": "asg-launch-template-2",
                "LaunchTemplateData": {
                    "ImageId": {
                        "Ref": "ECSAMI"
                    },
                    "InstanceType": {
```

```
                        "Ref": "InstanceType"
                    },
                    "IamInstanceProfile": {
                        "Name": {
                            "Ref": "EC2InstanceProfile"
                        }
                    },
                    "SecurityGroupIds": [
                        {
                            "Ref": "ContainerHostSecurityGroup"
                        }
                    ],
                    "UserData": {
                        "Fn::Base64": {
                            "Fn::Sub": "#!/bin/bash -xe\n echo ECS_CLUSTER=
${ECSCluster} >> /etc/ecs/ecs.config\n yum install -y aws-cfn-bootstrap\n /opt/aws/
bin/cfn-init -v --stack ${AWS::StackId} --resource ContainerInstances --configsets
 full_install --region ${AWS::Region} &\n"
                        }
                    },
                    "MetadataOptions": {
                        "HttpEndpoint": "enabled",
                        "HttpTokens": "required"
                    }
                }
            }
        },
        "EC2InstanceProfile": {
            "Type": "AWS::IAM::InstanceProfile",
            "Properties": {
                "Path": "/",
                "Roles": [
                    {
                        "Ref": "EC2Role"
                    }
                ]
            }
        },
        "CapacityProvider": {
            "Type": "AWS::ECS::CapacityProvider",
            "Properties": {
                "AutoScalingGroupProvider": {
                    "AutoScalingGroupArn": {
                        "Ref": "ECSAutoScalingGroup"
```

```
                    },
                    "ManagedScaling": {
                        "InstanceWarmupPeriod": 60,
                        "MinimumScalingStepSize": 1,
                        "MaximumScalingStepSize": 100,
                        "Status": "ENABLED",
                        "TargetCapacity": 100
                    },
                    "ManagedTerminationProtection": "ENABLED"
                }
            }
        },
        "CapacityProviderAssociation": {
            "Type": "AWS::ECS::ClusterCapacityProviderAssociations",
            "Properties": {
                "CapacityProviders": [
                    {
                        "Ref": "CapacityProvider"
                    }
                ],
                "Cluster": {
                    "Ref": "ECSCluster"
                },
                "DefaultCapacityProviderStrategy": [
                    {
                        "Base": 0,
                        "CapacityProvider": {
                            "Ref": "CapacityProvider"
                        },
                        "Weight": 1
                    }
                ]
            }
        },
        "ContainerHostSecurityGroup": {
            "Type": "AWS::EC2::SecurityGroup",
            "Properties": {
                "GroupDescription": "Access to the EC2 hosts that run containers",
                "VpcId": {
                    "Ref": "VpcId"
                }
            }
        },
        "EC2Role": {
```

```
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": [
                                    "ec2.amazonaws.com"
                                ]
                            },
                            "Action": [
                                "sts:AssumeRole"
                            ]
                        }
                    ]
                },
                "Path": "/",
                "ManagedPolicyArns": [
                    "arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role",
                    "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore"
                ]
            }
        },
        "ECSTaskExecutionRole": {
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": [
                                    "ecs-tasks.amazonaws.com"
                                ]
                            },
                            "Action": [
                                "sts:AssumeRole"
                            ],
                            "Condition": {
                                "ArnLike": {
                                    "aws:SourceArn": {
```

```
                                            "Fn::Sub": "arn:${AWS::Partition}:ecs:
${AWS::Region}:${AWS::AccountId}:*"
                                        }
                                    },
                                    "StringEquals": {
                                        "aws:SourceAccount": {
                                            "Fn::Sub": "${AWS::AccountId}"
                                        }
                                    }
                                }
                            }
                        ]
                    },
                    "Path": "/",
                    "ManagedPolicyArns": [
                        "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy"
                    ]
                }
            }
        },
        "Outputs": {
            "ClusterName": {
                "Description": "The ECS cluster into which to launch resources",
                "Value": "ECSCluster"
            },
            "ECSTaskExecutionRole": {
                "Description": "The role used to start up a task",
                "Value": "ECSTaskExecutionRole"
            },
            "CapacityProvider": {
                "Description": "The cluster capacity provider that the service should
 use to request capacity when it wants to start up a task",
                "Value": "CapacityProvider"
            }
        }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
```

```
Description: EC2 ECS cluster that starts out empty, with no EC2 instances yet. An
 ECS capacity provider automatically launches more EC2 instances as required on the
 fly when you request ECS to launch services or standalone tasks.
Parameters:
  InstanceType:
    Type: String
    Description: EC2 instance type
    Default: t2.medium
    AllowedValues:
      - t1.micro
      - t2.2xlarge
      - t2.large
      - t2.medium
      - t2.micro
      - t2.nano
      - t2.small
      - t2.xlarge
      - t3.2xlarge
      - t3.large
      - t3.medium
      - t3.micro
      - t3.nano
      - t3.small
      - t3.xlarge
  DesiredCapacity:
    Type: Number
    Default: '0'
    Description: Number of EC2 instances to launch in your ECS cluster.
  MaxSize:
    Type: Number
    Default: '100'
    Description: Maximum number of EC2 instances that can be launched in your ECS
 cluster.
  ECSAMI:
    Description: The Amazon Machine Image ID used for the cluster
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ecs/optimized-ami/amazon-linux-2023/recommended/image_id
  VpcId:
    Type: AWS::EC2::VPC::Id
    Description: VPC ID where the ECS cluster is launched
    Default: vpc-1234567890abcdef0
  SubnetIds:
    Type: List<AWS::EC2::Subnet::Id>
    Description: List of subnet IDs where the EC2 instances will be launched
```

```
    Default: subnet-021345abcdef67890
Resources:
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterSettings:
        - Name: containerInsights
          Value: enabled
  ECSAutoScalingGroup:
    Type: AWS::AutoScaling::AutoScalingGroup
    DependsOn:
      - ECSCluster
      - EC2Role
    Properties:
      VPCZoneIdentifier: !Ref SubnetIds
      LaunchTemplate:
        LaunchTemplateId: !Ref ContainerInstances
        Version: !GetAtt ContainerInstances.LatestVersionNumber
      MinSize: 0
      MaxSize: !Ref MaxSize
      DesiredCapacity: !Ref DesiredCapacity
      NewInstancesProtectedFromScaleIn: true
    UpdatePolicy:
      AutoScalingReplacingUpdate:
        WillReplace: 'true'
  ContainerInstances:
    Type: AWS::EC2::LaunchTemplate
    Properties:
      LaunchTemplateName: asg-launch-template-2
      LaunchTemplateData:
        ImageId: !Ref ECSAMI
        InstanceType: !Ref InstanceType
        IamInstanceProfile:
          Name: !Ref EC2InstanceProfile
        SecurityGroupIds:
          - !Ref ContainerHostSecurityGroup
        UserData: !Base64
          Fn::Sub: |
            #!/bin/bash -xe
            echo ECS_CLUSTER=${ECSCluster} >> /etc/ecs/ecs.config
            yum install -y aws-cfn-bootstrap
            /opt/aws/bin/cfn-init -v --stack ${AWS::StackId} --resource
 ContainerInstances --configsets full_install --region ${AWS::Region} &
        MetadataOptions:
```

```
          HttpEndpoint: enabled
          HttpTokens: required
  EC2InstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Path: /
      Roles:
        - !Ref EC2Role
  CapacityProvider:
    Type: AWS::ECS::CapacityProvider
    Properties:
      AutoScalingGroupProvider:
        AutoScalingGroupArn: !Ref ECSAutoScalingGroup
        ManagedScaling:
          InstanceWarmupPeriod: 60
          MinimumScalingStepSize: 1
          MaximumScalingStepSize: 100
          Status: ENABLED
          TargetCapacity: 100
        ManagedTerminationProtection: ENABLED
  CapacityProviderAssociation:
    Type: AWS::ECS::ClusterCapacityProviderAssociations
    Properties:
      CapacityProviders:
        - !Ref CapacityProvider
      Cluster: !Ref ECSCluster
      DefaultCapacityProviderStrategy:
        - Base: 0
          CapacityProvider: !Ref CapacityProvider
          Weight: 1
  ContainerHostSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Access to the EC2 hosts that run containers
      VpcId: !Ref VpcId
  EC2Role:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - ec2.amazonaws.com
```

```
            Action:
              - sts:AssumeRole
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role
        - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - ecs-tasks.amazonaws.com
            Action:
              - sts:AssumeRole
            Condition:
              ArnLike:
                aws:SourceArn: !Sub arn:${AWS::Partition}:ecs:${AWS::Region}:
${AWS::AccountId}:*
              StringEquals:
                aws:SourceAccount: !Sub ${AWS::AccountId}
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
Outputs:
  ClusterName:
    Description: The ECS cluster into which to launch resources
    Value: ECSCluster
  ECSTaskExecutionRole:
    Description: The role used to start up a task
    Value: ECSTaskExecutionRole
  CapacityProvider:
    Description: The cluster capacity provider that the service should use to
 request capacity when it wants to start up a task
    Value: CapacityProvider
```

Services

Vous pouvez utiliser un service Amazon ECS service pour exécuter et gérer simultanément un nombre spécifié d'instances d'une définition de tâche dans un cluster Amazon ECS. Si l'une de vos

tâches échoue ou s'arrête, le planificateur de service d'Amazon ECS service lance une autre instance de votre définition de tâche pour la remplacer. Cela permet de maintenir le nombre de tâches souhaité dans le service. Les modèles suivants peuvent être utilisés pour déployer des services. Pour plus d'informations sur les services Amazon ECS, consultezServices Amazon ECS.

Déployer une application

Le modèle suivant crée un service Amazon ECS nommé `cfn-service` qui lance et gère la tâche définie par la définition de tâche. Les modèles créent un cluster nommé`CFNCluster`. Le cluster contient une définition de tâche Linux Fargate qui configure un serveur Web. Le modèle crée également les ressources de mise en réseau et de journalisation nécessaires. Le modèle crée également un rôle d'exécution de tâche Amazon ECS. Pour plus d'informations sur le rôle d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "A template that deploys an application that is built on an
 Apache web server Docker image by creating an Amazon ECS cluster, task definition,
 and service. The template also creates networking and logging resources, and an
 Amazon ECS task execution role.",
    "Parameters": {
        "ClusterName": {
            "Type": "String",
            "Default": "CFNCluster",
            "Description": "Name of the ECS Cluster"
        },
        "TaskFamily": {
            "Type": "String",
            "Default": "task-definition-cfn",
            "Description": "Family name for the Task Definition"
        },
        "ServiceName": {
            "Type": "String",
            "Default": "cfn-service",
            "Description": "Name of the ECS Service"
        },
        "ContainerImage": {
            "Type": "String",
            "Default": "public.ecr.aws/docker/library/httpd:2.4",
            "Description": "Container image to use for the task"
```

```
        },
        "TaskCpu": {
            "Type": "Number",
            "Default": 256,
            "AllowedValues": [
                256,
                512,
                1024,
                2048,
                4096
            ],
            "Description": "CPU units for the task"
        },
        "TaskMemory": {
            "Type": "Number",
            "Default": 512,
            "AllowedValues": [
                512,
                1024,
                2048,
                4096,
                8192,
                16384
            ],
            "Description": "Memory (in MiB) for the task"
        },
        "DesiredCount": {
            "Type": "Number",
            "Default": 1,
            "Description": "Desired number of tasks to run"
        },
        "LogGroupName": {
            "Type": "String",
            "Default": "/ecs/fargate-task-definition",
            "Description": "CloudWatch Log Group name"
        },
        "VpcCidr": {
            "Type": "String",
            "Default": "10.0.0.0/16",
            "Description": "CIDR block for the VPC"
        },
        "PublicSubnet1Cidr": {
            "Type": "String",
            "Default": "10.0.0.0/24",
```

```
            "Description": "CIDR block for public subnet 1"
        },
        "PublicSubnet2Cidr": {
            "Type": "String",
            "Default": "10.0.1.0/24",
            "Description": "CIDR block for public subnet 2"
        }
    },
    "Resources": {
        "VPC": {
            "Type": "AWS::EC2::VPC",
            "Properties": {
                "CidrBlock": {
                    "Ref": "VpcCidr"
                },
                "EnableDnsSupport": true,
                "EnableDnsHostnames": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-VPC"
                        }
                    }
                ]
            }
        },
        "InternetGateway": {
            "Type": "AWS::EC2::InternetGateway",
            "Properties": {
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-IGW"
                        }
                    }
                ]
            }
        },
        "InternetGatewayAttachment": {
            "Type": "AWS::EC2::VPCGatewayAttachment",
            "Properties": {
                "InternetGatewayId": {
```

```
                    "Ref": "InternetGateway"
                },
                "VpcId": {
                    "Ref": "VPC"
                }
            }
        },
        "PublicSubnet1": {
            "Type": "AWS::EC2::Subnet",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "AvailabilityZone": {
                    "Fn::Select": [
                        0,
                        {
                            "Fn::GetAZs": ""
                        }
                    ]
                },
                "CidrBlock": {
                    "Ref": "PublicSubnet1Cidr"
                },
                "MapPublicIpOnLaunch": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-PublicSubnet1"
                        }
                    }
                ]
            }
        },
        "PublicSubnet2": {
            "Type": "AWS::EC2::Subnet",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "AvailabilityZone": {
                    "Fn::Select": [
                        1,
```

```
                        {
                            "Fn::GetAZs": ""
                        }
                    ]
                },
                "CidrBlock": {
                    "Ref": "PublicSubnet2Cidr"
                },
                "MapPublicIpOnLaunch": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-PublicSubnet2"
                        }
                    }
                ]
            }
        },
        "PublicRouteTable": {
            "Type": "AWS::EC2::RouteTable",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-PublicRouteTable"
                        }
                    }
                ]
            }
        },
        "DefaultPublicRoute": {
            "Type": "AWS::EC2::Route",
            "DependsOn": "InternetGatewayAttachment",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "DestinationCidrBlock": "0.0.0.0/0",
                "GatewayId": {
```

```
                    "Ref": "InternetGateway"
                }
            }
        },
        "PublicSubnet1RouteTableAssociation": {
            "Type": "AWS::EC2::SubnetRouteTableAssociation",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "SubnetId": {
                    "Ref": "PublicSubnet1"
                }
            }
        },
        "PublicSubnet2RouteTableAssociation": {
            "Type": "AWS::EC2::SubnetRouteTableAssociation",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "SubnetId": {
                    "Ref": "PublicSubnet2"
                }
            }
        },
        "ECSSecurityGroup": {
            "Type": "AWS::EC2::SecurityGroup",
            "Properties": {
                "GroupDescription": "Security group for ECS tasks",
                "VpcId": {
                    "Ref": "VPC"
                },
                "SecurityGroupIngress": [
                    {
                        "IpProtocol": "tcp",
                        "FromPort": 80,
                        "ToPort": 80,
                        "CidrIp": "0.0.0.0/0"
                    },
                    {
                        "IpProtocol": "tcp",
                        "FromPort": 443,
                        "ToPort": 443,
```

```
                        "CidrIp": "0.0.0.0/0"
                    }
                ]
            }
        },
        "ECSTaskExecutionRole": {
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": "ecs-tasks.amazonaws.com"
                            },
                            "Action": "sts:AssumeRole"
                        }
                    ]
                },
                "ManagedPolicyArns": [
                    "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy"
                ]
            }
        },
        "TaskLogGroup": {
            "Type": "AWS::Logs::LogGroup",
            "DeletionPolicy": "Retain",
            "UpdateReplacePolicy": "Retain",
            "Properties": {
                "LogGroupName": {
                    "Ref": "LogGroupName"
                },
                "RetentionInDays": 30
            }
        },
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": {
                    "Ref": "ClusterName"
                }
            }
```

```
        },
        "ECSTaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "ContainerDefinitions": [
                    {
                        "Command": [
                        "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
                        ],
                        "EntryPoint": [
                            "sh",
                            "-c"
                        ],
                        "Essential": true,
                        "Image": {
                            "Ref": "ContainerImage"
                        },
                        "LogConfiguration": {
                            "LogDriver": "awslogs",
                            "Options": {
                                "mode": "non-blocking",
                                "max-buffer-size": "25m",
                                "awslogs-create-group": "true",
                                "awslogs-group": {
                                    "Ref": "LogGroupName"
                                },
                                "awslogs-region": {
                                    "Ref": "AWS::Region"
                                },
                                "awslogs-stream-prefix": "ecs"
                            }
                        },
                        "Name": "sample-fargate-app",
                        "PortMappings": [
                            {
                                "ContainerPort": 80,
                                "HostPort": 80,
                                "Protocol": "tcp"
                            }
```

```
                        ]
                    }
                ],
                "Cpu": {
                    "Ref": "TaskCpu"
                },
                "ExecutionRoleArn": {
                    "Fn::GetAtt": [
                        "ECSTaskExecutionRole",
                        "Arn"
                    ]
                },
                "Family": {
                    "Ref": "TaskFamily"
                },
                "Memory": {
                    "Ref": "TaskMemory"
                },
                "NetworkMode": "awsvpc",
                "RequiresCompatibilities": [
                    "FARGATE"
                ],
                "RuntimePlatform": {
                    "OperatingSystemFamily": "LINUX"
                }
            }
        },
        "ECSService": {
            "Type": "AWS::ECS::Service",
            "DependsOn": [
                "PublicSubnet1RouteTableAssociation",
                "PublicSubnet2RouteTableAssociation"
            ],
            "Properties": {
                "ServiceName": {
                    "Ref": "ServiceName"
                },
                "Cluster": {
                    "Ref": "ECSCluster"
                },
                "DesiredCount": {
                    "Ref": "DesiredCount"
                },
                "LaunchType": "FARGATE",
```

```
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "AssignPublicIp": "ENABLED",
                        "SecurityGroups": [
                            {
                                "Ref": "ECSSecurityGroup"
                            }
                        ],
                        "Subnets": [
                            {
                                "Ref": "PublicSubnet1"
                            },
                            {
                                "Ref": "PublicSubnet2"
                            }
                        ]
                    }
                },
                "TaskDefinition": {
                    "Ref": "ECSTaskDefinition"
                }
            }
        }
    },
    "Outputs": {
        "ClusterName": {
            "Description": "The name of the ECS cluster",
            "Value": {
                "Ref": "ECSCluster"
            }
        },
        "TaskDefinitionArn": {
            "Description": "The ARN of the task definition",
            "Value": {
                "Ref": "ECSTaskDefinition"
            }
        },
        "ServiceName": {
            "Description": "The name of the ECS service",
            "Value": {
                "Ref": "ECSService"
            }
        },
        "VpcId": {
```

```
            "Description": "The ID of the VPC",
            "Value": {
                "Ref": "VPC"
            }
        },
        "PublicSubnet1": {
            "Description": "The ID of public subnet 1",
            "Value": {
                "Ref": "PublicSubnet1"
            }
        },
        "PublicSubnet2": {
            "Description": "The ID of public subnet 2",
            "Value": {
                "Ref": "PublicSubnet2"
            }
        },
        "SecurityGroup": {
            "Description": "The ID of the security group",
            "Value": {
                "Ref": "ECSSecurityGroup"
            }
        },
        "ExecutionRoleArn": {
            "Description": "The ARN of the task execution role",
            "Value": {
                "Fn::GetAtt": [
                    "ECSTaskExecutionRole",
                    "Arn"
                ]
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: A template that deploys an application that is built on an Apache
 web server Docker image by creating an Amazon ECS cluster, task definition, and
 service. The template also creates networking and logging resources, and an Amazon
 ECS task execution role.
Parameters:
```

```
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  TaskFamily:
    Type: String
    Default: task-definition-cfn
    Description: Family name for the Task Definition
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS Service
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  DesiredCount:
    Type: Number
    Default: 1
    Description: Desired number of tasks to run
  LogGroupName:
    Type: String
    Default: /ecs/fargate-task-definition
    Description: CloudWatch Log Group name
  VpcCidr:
    Type: String
    Default: 10.0.0.0/16
    Description: CIDR block for the VPC
  PublicSubnet1Cidr:
    Type: String
    Default: 10.0.0.0/24
    Description: CIDR block for public subnet 1
  PublicSubnet2Cidr:
    Type: String
```

```
    Default: 10.0.1.0/24
    Description: CIDR block for public subnet 2
Resources:
  # VPC and Networking Resources
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCidr
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-VPC
  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-IGW
  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC
  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PublicSubnet1Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet1
  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PublicSubnet2Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet2
```

```
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicRouteTable
  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1
  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2
  # Security Group
  ECSSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Security group for ECS tasks
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0

  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
```

```
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
  # CloudWatch Logs
  TaskLogGroup:
    Type: AWS::Logs::LogGroup
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      LogGroupName: !Ref LogGroupName
      RetentionInDays: 30
  # ECS Resources
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"s
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
```

```
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroupName
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: ecs
          Name: sample-fargate-app
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
      Cpu: !Ref TaskCpu
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      Family: !Ref TaskFamily
      Memory: !Ref TaskMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX
  ECSService:
    Type: AWS::ECS::Service
    DependsOn:
      - PublicSubnet1RouteTableAssociation
      - PublicSubnet2RouteTableAssociation
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ECSCluster
      DesiredCount: !Ref DesiredCount
      LaunchType: FARGATE
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
            - !Ref ECSSecurityGroup
          Subnets:
            - !Ref PublicSubnet1
            - !Ref PublicSubnet2
      TaskDefinition: !Ref ECSTaskDefinition
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  TaskDefinitionArn:
    Description: The ARN of the task definition
```

```
    Value: !Ref ECSTaskDefinition
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ECSService
  VpcId:
    Description: The ID of the VPC
    Value: !Ref VPC
  PublicSubnet1:
    Description: The ID of public subnet 1
    Value: !Ref PublicSubnet1
  PublicSubnet2:
    Description: The ID of public subnet 2
    Value: !Ref PublicSubnet2
  SecurityGroup:
    Description: The ID of the security group
    Value: !Ref ECSSecurityGroup
  ExecutionRoleArn:
    Description: The ARN of the task execution role
    Value: !GetAtt ECSTaskExecutionRole.Arn
```

Déployer un service avec ECS Exec activé

Vous pouvez utiliser le modèle suivant pour déployer un service avec ECS Exec activé. Le service s'exécute dans un cluster avec une clé KMS pour chiffrer les sessions ECS Exec et une configuration de journal pour rediriger les journaux des sessions de commande d'exécution vers un compartiment Amazon S3. Pour de plus amples informations, veuillez consulter Surveillez les conteneurs Amazon ECS avec ECS Exec.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "ECS Cluster with Fargate Service and Task Definition and ECS
 Exec enabled.",
    "Parameters": {
        "ClusterName": {
            "Type": "String",
            "Default": "CFNCluster",
            "Description": "Name of the ECS Cluster"
        },
        "S3BucketName": {
            "Type": "String",
```

```
            "Default": "amzn-s3-demo-bucket",
            "Description": "S3 bucket for ECS execute command logs"
        },
        "KmsKeyId": {
            "Type": "String",
            "Default": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Description": "KMS Key ID for ECS execute command encryption"
        },
        "ContainerImage": {
            "Type": "String",
            "Default": "public.ecr.aws/docker/library/httpd:2.4",
            "Description": "Container image to use for the task"
        },
        "ContainerCpu": {
            "Type": "Number",
            "Default": 256,
            "AllowedValues": [256, 512, 1024, 2048, 4096],
            "Description": "CPU units for the container (256 = 0.25 vCPU)"
        },
        "ContainerMemory": {
            "Type": "Number",
            "Default": 512,
            "AllowedValues": [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192],
            "Description": "Memory for the container (in MiB)"
        },
        "DesiredCount": {
            "Type": "Number",
            "Default": 1,
            "Description": "Desired count of tasks in the service"
        },
       "SecurityGroups": {
  "Type": "List<AWS::EC2::SecurityGroup::Id>",
  "Description": "Security Group IDs for the ECS Service"
 },
 "Subnets": {
  "Type": "List<AWS::EC2::Subnet::Id>",
  "Description": "Subnet IDs for the ECS Service"
 },
        "ServiceName": {
            "Type": "String",
            "Default": "cfn-service",
            "Description": "Name of the ECS service"
        },
        "TaskFamily": {
```

```
            "Type": "String",
            "Default": "task-definition-cfn",
            "Description": "Family name for the task definition"
        },
        "TaskExecutionRoleArn": {
            "Type": "String",
            "Description": "ARN of an existing IAM role for ECS task execution",
            "Default": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
        },
        "TaskRoleArn": {
            "Type": "String",
            "Description": "ARN of an existing IAM role for ECS tasks",
            "Default": "arn:aws:iam::111122223333:role/execTaskRole"
        }
    },
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": {"Ref": "ClusterName"},
                "Configuration": {
                    "ExecuteCommandConfiguration": {
                        "Logging": "OVERRIDE",
                        "LogConfiguration": {
                            "S3BucketName": {"Ref": "S3BucketName"}
                        },
                        "KmsKeyId": {"Ref": "KmsKeyId"}
                    }
                },
                "Tags": [
                    {
                        "Key": "Environment",
                        "Value": {"Ref": "AWS::StackName"}
                    }
                ]
            }
        },
        "ECSTaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "ContainerDefinitions": [
                    {
                        "Command": [
```

```
                                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS
 Sample App</title> <style>body {margin-top: 40px; background-color: #333;} </style>
 </head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
                        ],
                        "EntryPoint": [
                            "sh",
                            "-c"
                        ],
                        "Essential": true,
                        "Image": {"Ref": "ContainerImage"},
                        "LogConfiguration": {
                            "LogDriver": "awslogs",
                            "Options": {
                                "mode": "non-blocking",
                                "max-buffer-size": "25m",
                                "awslogs-create-group": "true",
                                "awslogs-group": {"Fn::Sub": "/ecs/
${AWS::StackName}"},
                                "awslogs-region": {"Ref": "AWS::Region"},
                                "awslogs-stream-prefix": "ecs"
                            }
                        },
                        "Name": "sample-fargate-app",
                        "PortMappings": [
                            {
                                "ContainerPort": 80,
                                "HostPort": 80,
                                "Protocol": "tcp"
                            }
                        ]
                    }
                ],
                "Cpu": {"Ref": "ContainerCpu"},
                "ExecutionRoleArn": {"Ref": "TaskExecutionRoleArn"},
                "TaskRoleArn": {"Ref": "TaskRoleArn"},
                "Family": {"Ref": "TaskFamily"},
                "Memory": {"Ref": "ContainerMemory"},
                "NetworkMode": "awsvpc",
                "RequiresCompatibilities": [
                    "FARGATE"
                ],
```

```
                "RuntimePlatform": {
                    "OperatingSystemFamily": "LINUX"
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {"Fn::Sub": "${AWS::StackName}-TaskDefinition"}
                    }
                ]
            }
        },
        "ECSService": {
            "Type": "AWS::ECS::Service",
            "Properties": {
                "ServiceName": {"Ref": "ServiceName"},
                "Cluster": {"Ref": "ECSCluster"},
                "DesiredCount": {"Ref": "DesiredCount"},
                "EnableExecuteCommand": true,
                "LaunchType": "FARGATE",
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "AssignPublicIp": "ENABLED",
                        "SecurityGroups": {"Ref": "SecurityGroups"},
                        "Subnets": {"Ref": "Subnets"}
                    }
                },
                "TaskDefinition": {"Ref": "ECSTaskDefinition"},
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {"Fn::Sub": "${AWS::StackName}-Service"}
                    }
                ]
            }
        }
    },
    "Outputs": {
        "ClusterName": {
            "Description": "The name of the ECS cluster",
            "Value": {"Ref": "ECSCluster"}
        },
        "ServiceName": {
            "Description": "The name of the ECS service",
            "Value": {"Ref": "ServiceName"}
```

```
        },
        "TaskDefinitionArn": {
            "Description": "The ARN of the task definition",
            "Value": {"Ref": "ECSTaskDefinition"}
        },
        "ClusterArn": {
            "Description": "The ARN of the ECS cluster",
            "Value": {"Fn::GetAtt": ["ECSCluster", "Arn"]}
        },
        "StackName": {
            "Description": "The name of this stack",
            "Value": {"Ref": "AWS::StackName"}
        },
        "StackId": {
            "Description": "The unique identifier for this stack",
            "Value": {"Ref": "AWS::StackId"}
        },
        "Region": {
            "Description": "The AWS Region where the stack is deployed",
            "Value": {"Ref": "AWS::Region"}
        },
        "AccountId": {
            "Description": "The AWS Account ID",
            "Value": {"Ref": "AWS::AccountId"}
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: ECS Cluster with Fargate Service and Task Definition and ECS Exec
 enabled.
Parameters:
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  S3BucketName:
    Type: String
    Default: amzn-s3-demo-bucket
    Description: S3 bucket for ECS execute command logs
  KmsKeyId:
```

```
    Type: String
    Default: a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
    Description: KMS Key ID for ECS execute command encryption
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  ContainerCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the container (256 = 0.25 vCPU)
  ContainerMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192]
    Description: Memory for the container (in MiB)
  DesiredCount:
    Type: Number
    Default: 1
    Description: Desired count of tasks in the service
 SecurityGroups:
    Type: List<AWS::EC2::SecurityGroup::Id>
    Description: Security Group IDs for the ECS Service
  Subnets:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Subnet IDs for the ECS Service
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS service
  TaskFamily:
    Type: String
    Default: task-definition-cfn
    Description: Family name for the task definition
  TaskExecutionRoleArn:
    Type: String
    Description: ARN of an existing IAM role for ECS task execution
    Default: 'arn:aws:iam::111122223333:role/ecsTaskExecutionRole'
  TaskRoleArn:
    Type: String
    Description: ARN of an existing IAM role for ECS tasks
    Default: 'arn:aws:iam::111122223333:role/execTaskRole'
Resources:
```

```
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
      Configuration:
        ExecuteCommandConfiguration:
          Logging: OVERRIDE
          LogConfiguration:
            S3BucketName: !Ref S3BucketName
          KmsKeyId: !Ref KmsKeyId
      Tags:
        - Key: Environment
          Value: !Ref AWS::StackName
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
             /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Sub /ecs/${AWS::StackName}
              awslogs-region: !Ref AWS::Region
              awslogs-stream-prefix: ecs
          Name: sample-fargate-app
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
```

```
                Protocol: tcp
      Cpu: !Ref ContainerCpu
      ExecutionRoleArn: !Ref TaskExecutionRoleArn
      TaskRoleArn: !Ref TaskRoleArn
      Family: !Ref TaskFamily
      Memory: !Ref ContainerMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-TaskDefinition
  ECSService:
    Type: AWS::ECS::Service
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ECSCluster
      DesiredCount: !Ref DesiredCount
      EnableExecuteCommand: true
      LaunchType: FARGATE
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups: !Ref SecurityGroups
          Subnets: !Ref Subnets
      TaskDefinition: !Ref ECSTaskDefinition
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-Service
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ServiceName
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref ECSTaskDefinition
  ClusterArn:
    Description: The ARN of the ECS cluster
    Value: !GetAtt ECSCluster.Arn
```

```
  StackName:
    Description: The name of this stack
    Value: !Ref AWS::StackName
  StackId:
    Description: The unique identifier for this stack
    Value: !Ref AWS::StackId
  Region:
    Description: The AWS Region where the stack is deployed
    Value: !Ref AWS::Region
  AccountId:
    Description: The AWS Account ID
    Value: !Ref AWS::AccountId
```

Déployer un service qui utilise Amazon VPC Lattice

Vous pouvez utiliser le modèle suivant pour commencer à créer un service Amazon ECS avec VPC Lattice. Vous devrez peut-être effectuer les étapes supplémentaires suivantes pour configurer VPC Lattice :

- Mettez à jour les règles entrantes de votre groupe de sécurité pour VPC Lattice afin d'autoriser le préfixe des `vpc-lattice` règles entrantes et d'autoriser le trafic sur le port 80.
- Associez le VPC du service à un réseau de services VPC Lattice.
- Configurez une zone hébergée privée ou publique avec Amazon Route 53.
- Configurez les écouteurs et les règles d'écoute dans un service VPC Lattice.
- Vérifiez les configurations de contrôle de santé du groupe cible.

Pour plus d'informations sur l'utilisation de VPC Lattice avec Amazon ECS, consultez. Utilisez Amazon VPC Lattice pour connecter, observer et sécuriser vos services Amazon ECS

JSON

```
{
 "AWSTemplateFormatVersion": "2010-09-09",
 "Description": "The template used to create an ECS Service with VPC Lattice.",
 "Parameters": {
  "ECSClusterName": {
   "Type": "String",
   "Default": "vpc-lattice-cluster"
  },
```

```
"ECSServiceName": {
 "Type": "String",
 "Default": "vpc-lattice-service"
},
"SecurityGroupIDs": {
 "Type": "List<AWS::EC2::SecurityGroup::Id>",
 "Description": "Security Group IDs for the ECS Service"
},
"SubnetIDs": {
 "Type": "List<AWS::EC2::Subnet::Id>",
 "Description": "Subnet IDs for the ECS Service"
},
"VpcID": {
 "Type": "AWS::EC2::VPC::Id",
 "Description": "VPC ID for the resources"
},
"ContainerImage": {
 "Type": "String",
 "Default": "public.ecr.aws/docker/library/httpd:2.4",
 "Description": "Container image to use for the task"
},
"TaskCpu": {
 "Type": "Number",
 "Default": 256,
 "AllowedValues": [256, 512, 1024, 2048, 4096],
 "Description": "CPU units for the task"
},
"TaskMemory": {
 "Type": "Number",
 "Default": 512,
 "AllowedValues": [512, 1024, 2048, 4096, 8192, 16384],
 "Description": "Memory (in MiB) for the task"
},
"LogGroupName": {
 "Type": "String",
 "Default": "/ecs/vpc-lattice-task",
 "Description": "CloudWatch Log Group name"
},
"EnableContainerInsights": {
 "Type": "String",
 "Default": "enabled",
 "AllowedValues": ["enabled", "disabled"],
 "Description": "Enable or disable CloudWatch Container Insights for the cluster"
}
```

```
  },
  "Resources": {
   "ECSCluster": {
    "Type": "AWS::ECS::Cluster",
    "Properties": {
     "ClusterName": {"Ref": "ECSClusterName"},
     "ClusterSettings": [
      {
       "Name": "containerInsights",
       "Value": {"Ref": "EnableContainerInsights"}
      }
     ],
     "Tags": [
      {
       "Key": "Name",
       "Value": {"Ref": "ECSClusterName"}
      }
     ]
    }
   },
   "ECSTaskExecutionRole": {
    "Type": "AWS::IAM::Role",
    "Properties": {
     "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
       {
        "Effect": "Allow",
        "Principal": {
         "Service": "ecs-tasks.amazonaws.com"
        },
        "Action": "sts:AssumeRole"
       }
      ]
     },
     "ManagedPolicyArns": [
      "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
     ]
    }
   },
   "TaskLogGroup": {
    "Type": "AWS::Logs::LogGroup",
    "DeletionPolicy": "Retain",
    "UpdateReplacePolicy": "Retain",
```

```
    "Properties": {
     "LogGroupName": {"Ref": "LogGroupName"},
     "RetentionInDays": 30
    }
   },
   "VpcLatticeTaskDefinition": {
    "Type": "AWS::ECS::TaskDefinition",
    "Properties": {
     "ContainerDefinitions": [
      {
       "Command": [
        "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</title>
<style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
<div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
<h2>Congratulations!</h2> <p>Your application is now running on a container in
Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
httpd-foreground\""
       ],
       "EntryPoint": [
        "sh",
        "-c"
       ],
       "Essential": true,
       "Image": {"Ref": "ContainerImage"},
       "LogConfiguration": {
        "LogDriver": "awslogs",
        "Options": {
         "mode": "non-blocking",
         "max-buffer-size": "25m",
         "awslogs-create-group": "true",
         "awslogs-group": {"Ref": "LogGroupName"},
         "awslogs-region": {"Ref": "AWS::Region"},
         "awslogs-stream-prefix": "ecs"
        }
       },
       "Name": "vpc-lattice-container",
       "PortMappings": [
        {
         "ContainerPort": 80,
         "HostPort": 80,
         "Protocol": "tcp",
         "Name": "vpc-lattice-port"
        }
       ]
```

```
      }
     ],
     "Cpu": {"Ref": "TaskCpu"},
     "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
     "Family": "vpc-lattice-task-definition",
     "Memory": {"Ref": "TaskMemory"},
     "NetworkMode": "awsvpc",
     "RequiresCompatibilities": [
      "FARGATE"
     ],
     "RuntimePlatform": {
      "OperatingSystemFamily": "LINUX"
     }
    }
   },
   "ECSService": {
    "Type": "AWS::ECS::Service",
    "Properties": {
     "Cluster": {"Ref": "ECSCluster"},
     "TaskDefinition": {"Ref": "VpcLatticeTaskDefinition"},
     "LaunchType": "FARGATE",
     "ServiceName": {"Ref": "ECSServiceName"},
     "SchedulingStrategy": "REPLICA",
     "DesiredCount": 2,
     "AvailabilityZoneRebalancing": "ENABLED",
     "NetworkConfiguration": {
      "AwsvpcConfiguration": {
       "AssignPublicIp": "ENABLED",
       "SecurityGroups": {
        "Ref": "SecurityGroupIDs"
       },
       "Subnets": {
        "Ref": "SubnetIDs"
       }
      }
     },
     "PlatformVersion": "LATEST",
     "VpcLatticeConfigurations": [
      {
       "RoleArn": "arn:aws:iam::111122223333:role/ecsInfrastructureRole",
       "PortName": "vpc-lattice-port",
       "TargetGroupArn": {
        "Ref": "TargetGroup1"
       }
```

```
      }
     ],
     "DeploymentConfiguration": {
      "DeploymentCircuitBreaker": {
       "Enable": true,
       "Rollback": true
      },
      "MaximumPercent": 200,
      "MinimumHealthyPercent": 100
     },
     "DeploymentController": {
      "Type": "ECS"
     },
     "ServiceConnectConfiguration": {
      "Enabled": false
     },
     "Tags": [],
     "EnableECSManagedTags": true
    }
   },
   "TargetGroup1": {
    "Type": "AWS::VpcLattice::TargetGroup",
    "Properties": {
     "Type": "IP",
     "Name": "first-target-group",
     "Config": {
      "Port": 80,
      "Protocol": "HTTP",
      "VpcIdentifier": {"Ref": "VpcID"},
      "HealthCheck": {
       "Enabled": true,
       "Path": "/"
      }
     },
     "Tags": [
      {
       "Key": "ecs-application-networking/ServiceName",
       "Value": {"Ref": "ECSServiceName"}
      },
      {
       "Key": "ecs-application-networking/ClusterName",
       "Value": {"Ref": "ECSClusterName"}
      },
      {
```

```
      "Key": "ecs-application-networking/TaskDefinition",
      "Value": {"Ref": "VpcLatticeTaskDefinition"}
     },
     {
      "Key": "ecs-application-networking/VpcId",
      "Value": {"Ref": "VpcID"}
     }
    ]
   }
  }
 },
 "Outputs": {
  "ClusterName": {
   "Description": "The cluster used to create the service.",
   "Value": {
    "Ref": "ECSCluster"
   }
  },
  "ClusterArn": {
   "Description": "The ARN of the ECS cluster",
   "Value": {
    "Fn::GetAtt": ["ECSCluster", "Arn"]
   }
  },
  "ECSService": {
   "Description": "The created service.",
   "Value": {
    "Ref": "ECSService"
   }
  },
  "TaskDefinitionArn": {
   "Description": "The ARN of the task definition",
   "Value": {
    "Ref": "VpcLatticeTaskDefinition"
   }
  }
 }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: The template used to create an ECS Service with VPC Lattice.
```

```
Parameters:
  ECSClusterName:
    Type: String
    Default: vpc-lattice-cluster
  ECSServiceName:
    Type: String
    Default: vpc-lattice-service
  SecurityGroupIDs:
    Type: List<AWS::EC2::SecurityGroup::Id>
    Description: Security Group IDs for the ECS Service
  SubnetIDs:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Subnet IDs for the ECS Service
  VpcID:
    Type: AWS::EC2::VPC::Id
    Description: VPC ID for the resources
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  LogGroupName:
    Type: String
    Default: /ecs/vpc-lattice-task
    Description: CloudWatch Log Group name
  EnableContainerInsights:
    Type: String
    Default: 'enhanced'
    AllowedValues: ['enabled', 'disabled', 'enhanced']
    Description: Enable or disable CloudWatch Container Insights for the cluster

Resources:
  # ECS Cluster
  ECSCluster:
```

```
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ECSClusterName
      ClusterSettings:
        - Name: containerInsights
          Value: !Ref EnableContainerInsights
      Tags:
        - Key: Name
          Value: !Ref ECSClusterName

  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy

  # CloudWatch Logs
  TaskLogGroup:
    Type: AWS::Logs::LogGroup
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      LogGroupName: !Ref LogGroupName
      RetentionInDays: 30

  # Task Definition
  VpcLatticeTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
```

```
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroupName
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: ecs
          Name: vpc-lattice-container
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
              Name: vpc-lattice-port
      Cpu: !Ref TaskCpu
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      Family: vpc-lattice-task-definition
      Memory: !Ref TaskMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX

  ECSService:
    Type: AWS::ECS::Service
    Properties:
      Cluster: !Ref ECSCluster
      TaskDefinition: !Ref VpcLatticeTaskDefinition
      LaunchType: FARGATE
      ServiceName: !Ref ECSServiceName
      SchedulingStrategy: REPLICA
      DesiredCount: 2
      AvailabilityZoneRebalancing: ENABLED
      NetworkConfiguration:
```

```
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups: !Ref SecurityGroupIDs
          Subnets: !Ref SubnetIDs
      PlatformVersion: LATEST
      VpcLatticeConfigurations:
        - RoleArn: arn:aws:iam::111122223333:role/ecsInfrastructureRole
          PortName: vpc-lattice-port
          TargetGroupArn: !Ref TargetGroup1
      DeploymentConfiguration:
        DeploymentCircuitBreaker:
          Enable: true
          Rollback: true
        MaximumPercent: 200
        MinimumHealthyPercent: 100
      DeploymentController:
        Type: ECS
      ServiceConnectConfiguration:
        Enabled: false
      Tags: []
      EnableECSManagedTags: true

  TargetGroup1:
    Type: AWS::VpcLattice::TargetGroup
    Properties:
      Type: IP
      Name: first-target-group
      Config:
        Port: 80
        Protocol: HTTP
        VpcIdentifier: !Ref VpcID
        HealthCheck:
          Enabled: true
          Path: /
      Tags:
        - Key: ecs-application-networking/ServiceName
          Value: !Ref ECSServiceName
        - Key: ecs-application-networking/ClusterName
          Value: !Ref ECSClusterName
        - Key: ecs-application-networking/TaskDefinition
          Value: !Ref VpcLatticeTaskDefinition
        - Key: ecs-application-networking/VpcId
          Value: !Ref VpcID
```

```
Outputs:
  ClusterName:
    Description: The cluster used to create the service.
    Value: !Ref ECSCluster
  ClusterArn:
    Description: The ARN of the ECS cluster
    Value: !GetAtt ECSCluster.Arn
  ECSService:
    Description: The created service.
    Value: !Ref ECSService
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref VpcLatticeTaskDefinition
```

Déployer le service avec une configuration en volume

Le modèle suivant inclut une configuration de volume dans la définition du service. Amazon ECS prend en charge la configuration des volumes de données suivants en utilisant une configuration de volume au lancement : volumes Amazon EBS. Pour plus d'informations sur les volumes Amazon EBS, reportez-vous à la section Utiliser des volumes Amazon EBS avec Amazon ECS.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "The template used to create an ECS Service that includes a
 volume configuration. The configuration is used to create Amazon EBS volumes for
 attachment to the tasks. One volume is attached per task.",
  "Parameters": {
    "ECSClusterName": {
      "Type": "String",
      "Default": "volume-config-cluster",
      "Description": "Name of the ECS cluster"
    },
    "SecurityGroupIDs": {
      "Type": "List<AWS::EC2::SecurityGroup::Id>",
      "Description": "Security Group IDs for the ECS Service"
    },
    "SubnetIDs": {
      "Type": "List<AWS::EC2::Subnet::Id>",
      "Description": "Subnet IDs for the ECS Service"
    },
```

```
    "InfrastructureRoleArn": {
      "Type": "String",
      "Description": "ARN of the IAM role that ECS will use to manage EBS volumes"
    },
    "ContainerImage": {
      "Type": "String",
      "Default": "public.ecr.aws/nginx/nginx:latest",
      "Description": "Container image to use for the task"
    },
    "TaskCpu": {
      "Type": "String",
      "Default": "2048",
      "Description": "CPU units for the task"
    },
    "TaskMemory": {
      "Type": "String",
      "Default": "4096",
      "Description": "Memory (in MiB) for the task"
    },
    "VolumeSize": {
      "Type": "String",
      "Default": "10",
      "Description": "Size of the EBS volume in GiB"
    },
    "VolumeType": {
      "Type": "String",
      "Default": "gp3",
      "AllowedValues": ["gp2", "gp3", "io1", "io2", "st1", "sc1", "standard"],
      "Description": "EBS volume type"
    },
    "VolumeIops": {
      "Type": "String",
      "Default": "3000",
      "Description": "IOPS for the EBS volume (required for io1, io2, and gp3)"
    },
    "VolumeThroughput": {
      "Type": "String",
      "Default": "125",
      "Description": "Throughput for the EBS volume (only for gp3)"
    },
    "FilesystemType": {
      "Type": "String",
      "Default": "xfs",
      "AllowedValues": ["xfs", "ext4"],
```

```
      "Description": "Filesystem type for the EBS volume"
    },
    "EnableContainerInsights": {
      "Type": "String",
      "Default": "enhanced",
      "AllowedValues": ["enabled", "disabled", "enhanced"],
      "Description": "Enable or disable CloudWatch Container Insights for the
cluster"
    }
  },
  "Resources": {
    "ECSCluster": {
      "Type": "AWS::ECS::Cluster",
      "Properties": {
        "ClusterName": {"Ref": "ECSClusterName"},
        "ClusterSettings": [
          {
            "Name": "containerInsights",
            "Value": {"Ref": "EnableContainerInsights"}
          }
        ],
        "Tags": [
          {
            "Key": "Name",
            "Value": {"Ref": "ECSClusterName"}
          }
        ]
      }
    },
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
```

```
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
      }
    },
    "EBSTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "Family": "ebs-task-attach-task-def",
        "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
        "NetworkMode": "awsvpc",
        "RequiresCompatibilities": [
          "EC2",
          "FARGATE"
        ],
        "Cpu": {"Ref": "TaskCpu"},
        "Memory": {"Ref": "TaskMemory"},
        "ContainerDefinitions": [
          {
            "Name": "nginx",
            "Image": {"Ref": "ContainerImage"},
            "Essential": true,
            "PortMappings": [
              {
                "Name": "nginx-80-tcp",
                "ContainerPort": 80,
                "HostPort": 80,
                "Protocol": "tcp",
                "AppProtocol": "http"
              }
            ],
            "MountPoints": [
              {
                "SourceVolume": "ebs-vol",
                "ContainerPath": "/foo-container-path",
                "ReadOnly": false
              }
            ]
          }
        ],
        "Volumes": [
          {
            "Name": "ebs-vol",
            "ConfiguredAtLaunch": true
```

```
          }
        ]
      }
    },
    "ECSService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": {"Ref": "ECSCluster"},
        "TaskDefinition": {"Ref": "EBSTaskDefinition"},
        "LaunchType": "FARGATE",
        "ServiceName": "ebs",
        "SchedulingStrategy": "REPLICA",
        "DesiredCount": 1,
        "NetworkConfiguration": {
          "AwsvpcConfiguration": {
            "AssignPublicIp": "ENABLED",
            "SecurityGroups": {"Ref": "SecurityGroupIDs"},
            "Subnets": {"Ref": "SubnetIDs"}
          }
        },
        "PlatformVersion": "LATEST",
        "DeploymentConfiguration": {
          "MaximumPercent": 200,
          "MinimumHealthyPercent": 100,
          "DeploymentCircuitBreaker": {
            "Enable": true,
            "Rollback": true
          }
        },
        "DeploymentController": {
          "Type": "ECS"
        },
        "Tags": [],
        "EnableECSManagedTags": true,
        "VolumeConfigurations": [
          {
            "Name": "ebs-vol",
            "ManagedEBSVolume": {
              "RoleArn": {"Ref": "InfrastructureRoleArn"},
              "VolumeType": {"Ref": "VolumeType"},
              "Iops": {"Ref": "VolumeIops"},
              "Throughput": {"Ref": "VolumeThroughput"},
              "SizeInGiB": {"Ref": "VolumeSize"},
              "FilesystemType": {"Ref": "FilesystemType"},
```

```
              "TagSpecifications": [
                {
                  "ResourceType": "volume",
                  "PropagateTags": "TASK_DEFINITION"
                }
              ]
            }
          }
        ]
      }
    }
  },
  "Outputs": {
    "ClusterName": {
      "Description": "The cluster used to create the service.",
      "Value": {"Ref": "ECSCluster"}
    },
    "ClusterArn": {
      "Description": "The ARN of the ECS cluster",
      "Value": {"Fn::GetAtt": ["ECSCluster", "Arn"]}
    },
    "ECSService": {
      "Description": "The created service.",
      "Value": {"Ref": "ECSService"}
    },
    "TaskDefinitionArn": {
      "Description": "The ARN of the task definition",
      "Value": {"Ref": "EBSTaskDefinition"}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: The template used to create an ECS Service that includes a volume
 configuration. The configuration is used to create Amazon EBS volumes for
 attachment to the tasks. One volume is attached per task.
Parameters:
  ECSClusterName:
    Type: String
    Default: volume-config-cluster
    Description: Name of the ECS cluster
```

```
  SecurityGroupIDs:
    Type: List<AWS::EC2::SecurityGroup::Id>
    Description: Security Group IDs for the ECS Service

  SubnetIDs:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Subnet IDs for the ECS Service

  InfrastructureRoleArn:
    Type: String
    Description: ARN of the IAM role that ECS will use to manage EBS volumes

  ContainerImage:
    Type: String
    Default: public.ecr.aws/nginx/nginx:latest
    Description: Container image to use for the task

  TaskCpu:
    Type: String
    Default: "2048"
    Description: CPU units for the task

  TaskMemory:
    Type: String
    Default: "4096"
    Description: Memory (in MiB) for the task

  VolumeSize:
    Type: String
    Default: "10"
    Description: Size of the EBS volume in GiB

  VolumeType:
    Type: String
    Default: gp3
    AllowedValues: [gp2, gp3, io1, io2, st1, sc1, standard]
    Description: EBS volume type

  VolumeIops:
    Type: String
    Default: "3000"
    Description: IOPS for the EBS volume (required for io1, io2, and gp3)
```

```
  VolumeThroughput:
    Type: String
    Default: "125"
    Description: Throughput for the EBS volume (only for gp3)

  FilesystemType:
    Type: String
    Default: xfs
    AllowedValues: [xfs, ext4]
    Description: Filesystem type for the EBS volume

  EnableContainerInsights:
    Type: String
    Default: 'enhanced'
    AllowedValues: ['enabled', 'disabled', 'enhanced']
    Description: Enable or disable CloudWatch Container Insights for the cluster

Resources:
  # ECS Cluster
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ECSClusterName
      ClusterSettings:
        - Name: containerInsights
          Value: !Ref EnableContainerInsights
      Tags:
        - Key: Name
          Value: !Ref ECSClusterName

  # IAM Role for Task Execution
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
```

```
  # Task Definition
  EBSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      Family: ebs-task-attach-task-def
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - EC2
        - FARGATE
      Cpu: !Ref TaskCpu
      Memory: !Ref TaskMemory
      ContainerDefinitions:
        - Name: nginx
          Image: !Ref ContainerImage
          Essential: true
          PortMappings:
            - Name: nginx-80-tcp
              ContainerPort: 80
              HostPort: 80
              Protocol: tcp
              AppProtocol: http
          MountPoints:
            - SourceVolume: ebs-vol
              ContainerPath: /foo-container-path
              ReadOnly: false
      Volumes:
        - Name: ebs-vol
          ConfiguredAtLaunch: true

  ECSService:
    Type: AWS::ECS::Service
    Properties:
      Cluster: !Ref ECSCluster
      TaskDefinition: !Ref EBSTaskDefinition
      LaunchType: FARGATE
      ServiceName: ebs
      SchedulingStrategy: REPLICA
      DesiredCount: 1
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups: !Ref SecurityGroupIDs
          Subnets: !Ref SubnetIDs
```

```
      PlatformVersion: LATEST
      DeploymentConfiguration:
        MaximumPercent: 200
        MinimumHealthyPercent: 100
        DeploymentCircuitBreaker:
          Enable: true
          Rollback: true
      DeploymentController:
        Type: ECS
      Tags: []
      EnableECSManagedTags: true
      VolumeConfigurations:
        - Name: ebs-vol
          ManagedEBSVolume:
            RoleArn: !Ref InfrastructureRoleArn
            VolumeType: !Ref VolumeType
            Iops: !Ref VolumeIops
            Throughput: !Ref VolumeThroughput
            SizeInGiB: !Ref VolumeSize
            FilesystemType: !Ref FilesystemType
            TagSpecifications:
              - ResourceType: volume
                PropagateTags: TASK_DEFINITION

Outputs:
  ClusterName:
    Description: The cluster used to create the service.
    Value: !Ref ECSCluster
  ClusterArn:
    Description: The ARN of the ECS cluster
    Value: !GetAtt ECSCluster.Arn
  ECSService:
    Description: The created service.
    Value: !Ref ECSService
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref EBSTaskDefinition
```

Déployer le service auprès de fournisseurs de capacité

Le modèle suivant définit un service qui utilise le fournisseur de capacité pour demander une capacité AL2 023 à exécuter. Les conteneurs seront lancés sur les instances AL2 023 au fur et à mesure de leur mise en ligne.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "An example service that deploys in AWS VPC networking mode on
 EC2 capacity. Service uses a capacity provider to request EC2 instances to run
 on. Service runs with networking in private subnets, but still accessible to the
 internet via a load balancer hosted in public subnets.",
  "Parameters": {
      "VpcId": {
          "Type": "String",
          "Description": "The VPC that the service is running inside of"
      },
      "PublicSubnetIds": {
          "Type": "List<AWS::EC2::Subnet::Id>",
          "Description": "List of public subnet ID's to put the load balancer in"
      },
      "PrivateSubnetIds": {
          "Type": "List<AWS::EC2::Subnet::Id>",
          "Description": "List of private subnet ID's that the AWS VPC tasks are in"
      },
      "ClusterName": {
          "Type": "String",
          "Description": "The name of the ECS cluster into which to launch
 capacity."
      },
      "ECSTaskExecutionRole": {
          "Type": "String",
          "Description": "The role used to start up an ECS task"
      },
      "CapacityProvider": {
          "Type": "String",
          "Description": "The cluster capacity provider that the service should use
 to request capacity when it wants to start up a task"
      },
      "ServiceName": {
          "Type": "String",
```

```
            "Default": "web",
            "Description": "A name for the service"
        },
        "ImageUrl": {
            "Type": "String",
            "Default": "public.ecr.aws/docker/library/nginx:latest",
            "Description": "The url of a docker image that contains the application
process that will handle the traffic for this service"
        },
        "ContainerCpu": {
            "Type": "Number",
            "Default": 256,
            "Description": "How much CPU to give the container. 1024 is 1 CPU"
        },
        "ContainerMemory": {
            "Type": "Number",
            "Default": 512,
            "Description": "How much memory in megabytes to give the container"
        },
        "ContainerPort": {
            "Type": "Number",
            "Default": 80,
            "Description": "What port that the application expects traffic on"
        },
        "DesiredCount": {
            "Type": "Number",
            "Default": 2,
            "Description": "How many copies of the service task to run"
        }
    },
    "Resources": {
        "TaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "Family": {
                    "Ref": "ServiceName"
                },
                "Cpu": {
                    "Ref": "ContainerCpu"
                },
                "Memory": {
                    "Ref": "ContainerMemory"
                },
                "NetworkMode": "awsvpc",
```

```
                "RequiresCompatibilities": [
                    "EC2"
                ],
                "ExecutionRoleArn": {
                    "Ref": "ECSTaskExecutionRole"
                },
                "ContainerDefinitions": [
                    {
                        "Name": {
                            "Ref": "ServiceName"
                        },
                        "Cpu": {
                            "Ref": "ContainerCpu"
                        },
                        "Memory": {
                            "Ref": "ContainerMemory"
                        },
                        "Image": {
                            "Ref": "ImageUrl"
                        },
                        "PortMappings": [
                            {
                                "ContainerPort": {
                                    "Ref": "ContainerPort"
                                },
                                "HostPort": {
                                    "Ref": "ContainerPort"
                                }
                            }
                        ],
                        "LogConfiguration": {
                            "LogDriver": "awslogs",
                            "Options": {
                                "mode": "non-blocking",
                                "max-buffer-size": "25m",
                                "awslogs-group": {
                                    "Ref": "LogGroup"
                                },
                                "awslogs-region": {
                                    "Ref": "AWS::Region"
                                },
                                "awslogs-stream-prefix": {
                                    "Ref": "ServiceName"
                                }
```

```
                        }
                    }
                }
            ]
        }
    },
    "Service": {
        "Type": "AWS::ECS::Service",
        "DependsOn": "PublicLoadBalancerListener",
        "Properties": {
            "ServiceName": {
                "Ref": "ServiceName"
            },
            "Cluster": {
                "Ref": "ClusterName"
            },
            "PlacementStrategies": [
                {
                    "Field": "attribute:ecs.availability-zone",
                    "Type": "spread"
                },
                {
                    "Field": "cpu",
                    "Type": "binpack"
                }
            ],
            "CapacityProviderStrategy": [
                {
                    "Base": 0,
                    "CapacityProvider": {
                        "Ref": "CapacityProvider"
                    },
                    "Weight": 1
                }
            ],
            "NetworkConfiguration": {
                "AwsvpcConfiguration": {
                    "SecurityGroups": [
                        {
                            "Ref": "ServiceSecurityGroup"
                        }
                    ],
                    "Subnets": {
                        "Ref": "PrivateSubnetIds"
```

```
                    }
                }
            },
            "DeploymentConfiguration": {
                "MaximumPercent": 200,
                "MinimumHealthyPercent": 75
            },
            "DesiredCount": {
                "Ref": "DesiredCount"
            },
            "TaskDefinition": {
                "Ref": "TaskDefinition"
            },
            "LoadBalancers": [
                {
                    "ContainerName": {
                        "Ref": "ServiceName"
                    },
                    "ContainerPort": {
                        "Ref": "ContainerPort"
                    },
                    "TargetGroupArn": {
                        "Ref": "ServiceTargetGroup"
                    }
                }
            ]
        }
    },
    "ServiceSecurityGroup": {
        "Type": "AWS::EC2::SecurityGroup",
        "Properties": {
            "GroupDescription": "Security group for service",
            "VpcId": {
                "Ref": "VpcId"
            }
        }
    },
    "ServiceTargetGroup": {
        "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
        "Properties": {
            "HealthCheckIntervalSeconds": 6,
            "HealthCheckPath": "/",
            "HealthCheckProtocol": "HTTP",
            "HealthCheckTimeoutSeconds": 5,
```

```
                "HealthyThresholdCount": 2,
                "TargetType": "ip",
                "Port": {
                    "Ref": "ContainerPort"
                },
                "Protocol": "HTTP",
                "UnhealthyThresholdCount": 10,
                "VpcId": {
                    "Ref": "VpcId"
                },
                "TargetGroupAttributes": [
                    {
                        "Key": "deregistration_delay.timeout_seconds",
                        "Value": 0
                    }
                ]
            }
        },
        "PublicLoadBalancerSG": {
            "Type": "AWS::EC2::SecurityGroup",
            "Properties": {
                "GroupDescription": "Access to the public facing load balancer",
                "VpcId": {
                    "Ref": "VpcId"
                },
                "SecurityGroupIngress": [
                    {
                        "CidrIp": "0.0.0.0/0",
                        "IpProtocol": -1
                    }
                ]
            }
        },
        "PublicLoadBalancer": {
            "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
            "Properties": {
                "Scheme": "internet-facing",
                "LoadBalancerAttributes": [
                    {
                        "Key": "idle_timeout.timeout_seconds",
                        "Value": "30"
                    }
                ],
                "Subnets": {
```

```
                "Ref": "PublicSubnetIds"
            },
            "SecurityGroups": [
                {
                    "Ref": "PublicLoadBalancerSG"
                }
            ]
        }
    },
    "PublicLoadBalancerListener": {
        "Type": "AWS::ElasticLoadBalancingV2::Listener",
        "Properties": {
            "DefaultActions": [
                {
                    "Type": "forward",
                    "ForwardConfig": {
                        "TargetGroups": [
                            {
                                "TargetGroupArn": {
                                    "Ref": "ServiceTargetGroup"
                                },
                                "Weight": 100
                            }
                        ]
                    }
                }
            ],
            "LoadBalancerArn": {
                "Ref": "PublicLoadBalancer"
            },
            "Port": 80,
            "Protocol": "HTTP"
        }
    },
    "ServiceIngressfromLoadBalancer": {
        "Type": "AWS::EC2::SecurityGroupIngress",
        "Properties": {
            "Description": "Ingress from the public ALB",
            "GroupId": {
                "Ref": "ServiceSecurityGroup"
            },
            "IpProtocol": -1,
            "SourceSecurityGroupId": {
                "Ref": "PublicLoadBalancerSG"
```

```
                }
            }
        },
        "LogGroup": {
            "Type": "AWS::Logs::LogGroup"
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: >-
  An example service that deploys in AWS VPC networking mode on EC2 capacity.
  Service uses a capacity provider to request EC2 instances to run on. Service
  runs with networking in private subnets, but still accessible to the internet
  via a load balancer hosted in public subnets.
Parameters:
  VpcId:
    Type: String
    Description: The VPC that the service is running inside of
  PublicSubnetIds:
    Type: 'List<AWS::EC2::Subnet::Id>'
    Description: List of public subnet ID's to put the load balancer in
  PrivateSubnetIds:
    Type: 'List<AWS::EC2::Subnet::Id>'
    Description: List of private subnet ID's that the AWS VPC tasks are in
  ClusterName:
    Type: String
    Description: The name of the ECS cluster into which to launch capacity.
  ECSTaskExecutionRole:
    Type: String
    Description: The role used to start up an ECS task
  CapacityProvider:
    Type: String
    Description: >-
      The cluster capacity provider that the service should use to request
      capacity when it wants to start up a task
  ServiceName:
    Type: String
    Default: web
    Description: A name for the service
  ImageUrl:
```

```
    Type: String
    Default: 'public.ecr.aws/docker/library/nginx:latest'
    Description: >-
      The url of a docker image that contains the application process that will
      handle the traffic for this service
  ContainerCpu:
    Type: Number
    Default: 256
    Description: How much CPU to give the container. 1024 is 1 CPU
  ContainerMemory:
    Type: Number
    Default: 512
    Description: How much memory in megabytes to give the container
  ContainerPort:
    Type: Number
    Default: 80
    Description: What port that the application expects traffic on
  DesiredCount:
    Type: Number
    Default: 2
    Description: How many copies of the service task to run
Resources:
  TaskDefinition:
    Type: 'AWS::ECS::TaskDefinition'
    Properties:
      Family: !Ref ServiceName
      Cpu: !Ref ContainerCpu
      Memory: !Ref ContainerMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - EC2
      ExecutionRoleArn: !Ref ECSTaskExecutionRole
      ContainerDefinitions:
        - Name: !Ref ServiceName
          Cpu: !Ref ContainerCpu
          Memory: !Ref ContainerMemory
          Image: !Ref ImageUrl
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              HostPort: !Ref ContainerPort
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
```

```
            max-buffer-size: 25m
            awslogs-group: !Ref LogGroup
            awslogs-region: !Ref AWS::Region
            awslogs-stream-prefix: !Ref ServiceName
  Service:
    Type: AWS::ECS::Service
    DependsOn: PublicLoadBalancerListener
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ClusterName
      PlacementStrategies:
        - Field: 'attribute:ecs.availability-zone'
          Type: spread
        - Field: cpu
          Type: binpack
      CapacityProviderStrategy:
        - Base: 0
          CapacityProvider: !Ref CapacityProvider
          Weight: 1
      NetworkConfiguration:
        AwsvpcConfiguration:
          SecurityGroups:
            - !Ref ServiceSecurityGroup
          Subnets: !Ref PrivateSubnetIds
      DeploymentConfiguration:
        MaximumPercent: 200
        MinimumHealthyPercent: 75
      DesiredCount: !Ref DesiredCount
      TaskDefinition: !Ref TaskDefinition
      LoadBalancers:
        - ContainerName: !Ref ServiceName
          ContainerPort: !Ref ContainerPort
          TargetGroupArn: !Ref ServiceTargetGroup
  ServiceSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Security group for service
      VpcId: !Ref VpcId
  ServiceTargetGroup:
    Type: 'AWS::ElasticLoadBalancingV2::TargetGroup'
    Properties:
      HealthCheckIntervalSeconds: 6
      HealthCheckPath: /
      HealthCheckProtocol: HTTP
```

```
      HealthCheckTimeoutSeconds: 5
      HealthyThresholdCount: 2
      TargetType: ip
      Port: !Ref ContainerPort
      Protocol: HTTP
      UnhealthyThresholdCount: 10
      VpcId: !Ref VpcId
      TargetGroupAttributes:
        - Key: deregistration_delay.timeout_seconds
          Value: 0
  PublicLoadBalancerSG:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: Access to the public facing load balancer
      VpcId: !Ref VpcId
      SecurityGroupIngress:
        - CidrIp: 0.0.0.0/0
          IpProtocol: -1
  PublicLoadBalancer:
    Type: 'AWS::ElasticLoadBalancingV2::LoadBalancer'
    Properties:
      Scheme: internet-facing
      LoadBalancerAttributes:
        - Key: idle_timeout.timeout_seconds
          Value: '30'
      Subnets: !Ref PublicSubnetIds
      SecurityGroups:
        - !Ref PublicLoadBalancerSG
  PublicLoadBalancerListener:
    Type: 'AWS::ElasticLoadBalancingV2::Listener'
    Properties:
      DefaultActions:
        - Type: forward
          ForwardConfig:
            TargetGroups:
              - TargetGroupArn: !Ref ServiceTargetGroup
                Weight: 100
      LoadBalancerArn: !Ref PublicLoadBalancer
      Port: 80
      Protocol: HTTP
  ServiceIngressfromLoadBalancer:
    Type: 'AWS::EC2::SecurityGroupIngress'
    Properties:
      Description: Ingress from the public ALB
```

```
      GroupId: !Ref ServiceSecurityGroup
      IpProtocol: -1
      SourceSecurityGroupId: !Ref PublicLoadBalancerSG
  LogGroup:
    Type: 'AWS::Logs::LogGroup'
```

Rôles IAM pour Amazon ECS

Vous pouvez utiliser des AWS CloudFormation modèles pour créer des rôles IAM à utiliser avec Amazon ECS. Pour plus d'informations sur les rôles IAM pour Amazon ECS, consultez Rôles IAM pour Amazon ECS.

Rôle d'exécution des tâches Amazon ECS

Le rôle d'exécution des tâches autorise le conteneur Amazon ECS et les agents Fargate à AWS effectuer des appels d'API en votre nom. Le rôle est requis en fonction des exigences de votre tâche. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

Le modèle suivant peut être utilisé pour créer un rôle d'exécution de tâche simple qui utilise la politique `AmazonECSTaskExecutionRolePolicy` gérée.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "CloudFormation template for ECS Task Execution Role",
  "Resources": {
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": ["ecs-tasks.amazonaws.com"]
              },
              "Action": ["sts:AssumeRole"],
              "Condition": {
                "ArnLike": {
                  "aws:SourceArn": {
```

```
                    "Fn::Sub": "arn:aws:ecs:${AWS::Region}:${AWS::AccountId}:*"
                  }
                },
                "StringEquals": {
                  "aws:SourceAccount": {
                    "Ref": "AWS::AccountId"
                  }
                }
              }
            }
          ]
        },
        "Path": "/",
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
      }
    }
  },
  "Outputs": {
    "ECSTaskExecutionRoleARN": {
      "Description": "ARN of the ECS Task Execution Role",
      "Value": {
        "Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]
      },
      "Export": {
        "Name": {
          "Fn::Sub": "${AWS::StackName}-ECSTaskExecutionRoleARN"
        }
      }
    },
    "ECSTaskExecutionRoleName": {
      "Description": "Name of the ECS Task Execution Role",
      "Value": {
        "Ref": "ECSTaskExecutionRole"
      },
      "Export": {
        "Name": {
          "Fn::Sub": "${AWS::StackName}-ECSTaskExecutionRoleName"
        }
      }
    }
  }
```

```
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: 'CloudFormation template for ECS Task Execution Role'
Resources:
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Principal:
              Service: [ecs-tasks.amazonaws.com]
            Action: ['sts:AssumeRole']
            Condition:
              ArnLike:
                aws:SourceArn: !Sub arn:aws:ecs:${AWS::Region}:${AWS::AccountId}:*
              StringEquals:
                aws:SourceAccount: !Ref AWS::AccountId
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
Outputs:
  ECSTaskExecutionRoleARN:
    Description: ARN of the ECS Task Execution Role
    Value: !GetAtt ECSTaskExecutionRole.Arn
    Export:
      Name: !Sub "${AWS::StackName}-ECSTaskExecutionRoleARN"
  ECSTaskExecutionRoleName:
    Description: Name of the ECS Task Execution Role
    Value: !Ref ECSTaskExecutionRole
    Export:
      Name: !Sub "${AWS::StackName}-ECSTaskExecutionRoleName"
```

Création de ressources Amazon ECS à l'aide de la AWS CloudFormation console

L'une des manières d'utiliser Amazon ECS AWS CloudFormation est d'utiliser le AWS Management Console. Ici, vous pouvez créer vos AWS CloudFormation piles pour les composants Amazon ECS

tels que les définitions de tâches, les clusters et les services et les déployer directement depuis la console. Le didacticiel suivant montre comment utiliser la AWS CloudFormation console pour créer un service Amazon ECS, une définition de tâche et un cluster.

Prérequis

Ce didacticiel part du principe que les prérequis suivants sont remplis.

- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique Amazon ECS_ FullAccess IAM.

Étape 1 : créer un modèle de pile

Suivez les étapes ci-dessous pour créer un modèle de AWS CloudFormation pile pour un service Amazon ECS et d'autres ressources associées.

1. À l'aide de l'éditeur de texte de votre choix, créez un fichier appelé`ecs-tutorial-template.yaml`.
2. Dans le `ecs-tutorial-template.yaml` fichier, collez le modèle suivant et enregistrez les modifications.

```
AWSTemplateFormatVersion: 2010-09-09
Description: A template that deploys an application that is built on an Apache
 web server Docker image by creating an Amazon ECS cluster, task definition, and
 service. The template also creates networking and logging resources, and an Amazon
 ECS task execution role.
Parameters:
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  TaskFamily:
    Type: String
    Default: task-definition-cfn
    Description: Family name for the Task Definition
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS Service
```

```
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  DesiredCount:
    Type: Number
    Default: 1
    Description: Desired number of tasks to run
  LogGroupName:
    Type: String
    Default: /ecs/fargate-task-definition
    Description: CloudWatch Log Group name
  VpcCidr:
    Type: String
    Default: 10.0.0.0/16
    Description: CIDR block for the VPC
  PublicSubnet1Cidr:
    Type: String
    Default: 10.0.0.0/24
    Description: CIDR block for public subnet 1
  PublicSubnet2Cidr:
    Type: String
    Default: 10.0.1.0/24
    Description: CIDR block for public subnet 2
Resources:
  # VPC and Networking Resources
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCidr
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
```

```
        Value: !Sub ${AWS::StackName}-VPC
  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-IGW
  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC
  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PublicSubnet1Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet1
  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PublicSubnet2Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet2
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicRouteTable
  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
```

```
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1
  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2
  # Security Group
  ECSSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Security group for ECS tasks
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0
  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
  # CloudWatch Logs
  TaskLogGroup:
    Type: AWS::Logs::LogGroup
    DeletionPolicy: Retain
```

```
    UpdateReplacePolicy: Retain
    Properties:
      LogGroupName: !Ref LogGroupName
      RetentionInDays: 30
  # ECS Resources
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"s
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroupName
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: ecs
          Name: sample-fargate-app
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
              Protocol: tcp
      Cpu: !Ref TaskCpu
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      Family: !Ref TaskFamily
```

```
      Memory: !Ref TaskMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX
  ECSService:
    Type: AWS::ECS::Service
    DependsOn:
      - PublicSubnet1RouteTableAssociation
      - PublicSubnet2RouteTableAssociation
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ECSCluster
      DesiredCount: !Ref DesiredCount
      LaunchType: FARGATE
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
            - !Ref ECSSecurityGroup
          Subnets:
            - !Ref PublicSubnet1
            - !Ref PublicSubnet2
      TaskDefinition: !Ref ECSTaskDefinition
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref ECSTaskDefinition
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ECSService
  VpcId:
    Description: The ID of the VPC
    Value: !Ref VPC
  PublicSubnet1:
    Description: The ID of public subnet 1
    Value: !Ref PublicSubnet1
  PublicSubnet2:
    Description: The ID of public subnet 2
    Value: !Ref PublicSubnet2
```

```
  SecurityGroup:
    Description: The ID of the security group
    Value: !Ref ECSSecurityGroup
  ExecutionRoleArn:
    Description: The ARN of the task execution role
    Value: !GetAtt ECSTaskExecutionRole.Arn
```

Étape 2 : créer une pile pour les ressources Amazon ECS

Après avoir créé un fichier pour le modèle, vous pouvez suivre ces étapes pour créer une pile avec le modèle à l'aide de la AWS CloudFormation console.

1. Connectez-vous à la AWS CloudFormation console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/cloudformation.
2. Sur la page Stacks, dans le coin supérieur droit, choisissez Create stack, puis choisissez avec de nouvelles ressources (standard).
3. Choisissez Choisir un modèle existant.
4. Choisissez Télécharger un fichier modèle, puis choisissez Choisir un fichier pour sélectionner le `ecs-tutorial-template` fichier.

   Une fois le fichier chargé dans un compartiment Amazon S3, vous pouvez choisir Afficher dans Infrastructure Composer pour visualiser le modèle dans Infrastructure Composer. Pour plus d'informations sur les AWS CloudFormation modèles et Infrastructure Composer, voir Création visuelle de modèles avec Infrastructure Composer dans le guide de AWS CloudFormation l'utilisateur.
5. Choisissez Suivant.
6. Sur la page Spécifier les détails de la pile, sous Nom de la pile, saisissez le nom suivant pour la pile :`ecs-tutorial-stack`. Laissez toutes les valeurs des paramètres sous Paramètres comme valeurs par défaut, puis choisissez Next.
7. Sur la page Configurer les options de pile, sous Fonctionnalités, cochez la case pour confirmer la AWS CloudFormation création de ressources IAM. Cet accusé de réception est nécessaire pour créer le rôle d'exécution des tâches Amazon ECS tel que défini dans le modèle. Conservez les autres paramètres par défaut et choisissez Next.
8. Vérifiez les détails de la pile sur la page Révision et création, puis choisissez Soumettre pour lancer la création de la pile.

Étape 3 : vérifier

Suivez les étapes suivantes pour vérifier la création de ressources Amazon ECS à l'aide du modèle fourni.

1. Connectez-vous à la AWS CloudFormation console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/cloudformation.
2. Sur la page Stacks, choisissez ecs-tutorial-stack.
3. Choisissez l'onglet Événements. Si le statut de l'événement indique`CREATE_IN_PROGRESS`, attendez que la création soit terminée et que les statuts passent à. `CREATE_COMPLETE`
4. Une fois que les statuts des événements sont passés à`CREATE_COMPLETE`, choisissez l'onglet Ressources. Vous verrez les ressources avec l'ID logique `ECSClusterECSTaskDefinition`, et `ECSService` respectivement.
5. Pour vérifier la création d'un cluster Amazon ECS, choisissez l'ID physique associé à`ECSCluster`. Vous serez redirigé vers la console Amazon ECS où vous pourrez voir le cluster créé appelé`CFNCluster`.
6. Pour vérifier la création d'un service Amazon ECS, choisissez l'ID physique associé à`ECSService`. Vous serez redirigé vers la console Amazon ECS où vous pourrez voir le service appelé `cfn-service` créé dans le cluster`cfnCluster`.
7. Pour vérifier la création d'une définition de tâche Amazon ECS, choisissez l'ID physique associé à`ECSTaskDefinition`. Vous serez redirigé vers la console Amazon ECS où vous pourrez voir la révision de la définition de la tâche avec le nom`task-definition-cfn`.

Étape 4 : Nettoyer les ressources

Pour nettoyer les ressources et éviter d'encourir des coûts supplémentaires, procédez comme suit.

1. Connectez-vous à la AWS CloudFormation console AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/cloudformation.
2. Sur la page Stacks, choisissez ecs-tutorial-stack.
3. Sélectionnez Delete (Supprimer).
4. Lorsque vous êtes invité à confirmer, sélectionnez à nouveau Supprimer.
5. Choisissez l'onglet Événements. État des `ecs-tutorial-stack` modifications apportées à DELETE_IN_PROGRESS puis à DELETE_COMPLETE après la suppression ou le désenregistrement des ressources. La suppression prend quelques minutes.

6. Sélectionnez l'onglet Ressources. Vous verrez maintenant une liste d'identifiants logiques dont le statut a été mis à jour en DELETE_COMPLETE.

Création de ressources Amazon ECS à l'aide de AWS CLI commandes pour AWS CloudFormation

Le didacticiel suivant montre comment vous pouvez utiliser le AWS CLI pour créer des ressources Amazon ECS à l'aide d'un AWS CloudFormation modèle.

Prérequis

- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique Amazon ECS_ FullAccess IAM.

Étape 1 : créer une pile

Pour créer une pile à l'aide du AWS CLI, procédez comme suit.

1. À l'aide de l'éditeur de texte de votre choix, créez un fichier appelé`ecs-tutorial-template.yaml`.
2. Dans le `ecs-tutorial-template.yaml` fichier, collez le modèle suivant et enregistrez les modifications.

```
AWSTemplateFormatVersion: 2010-09-09
Description: A template that deploys an application that is built on an Apache
 web server Docker image by creating an Amazon ECS cluster, task definition, and
 service. The template also creates networking and logging resources, and an Amazon
 ECS task execution role.
Parameters:
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  TaskFamily:
    Type: String
    Default: task-definition-cfn
    Description: Family name for the Task Definition
```

```
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS Service
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  DesiredCount:
    Type: Number
    Default: 1
    Description: Desired number of tasks to run
  LogGroupName:
    Type: String
    Default: /ecs/fargate-task-definition
    Description: CloudWatch Log Group name
  VpcCidr:
    Type: String
    Default: 10.0.0.0/16
    Description: CIDR block for the VPC
  PublicSubnet1Cidr:
    Type: String
    Default: 10.0.0.0/24
    Description: CIDR block for public subnet 1
  PublicSubnet2Cidr:
    Type: String
    Default: 10.0.1.0/24
    Description: CIDR block for public subnet 2
Resources:
  # VPC and Networking Resources
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCidr
```

```
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-VPC
  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-IGW
  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC
  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PublicSubnet1Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet1
  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PublicSubnet2Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicSubnet2
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-PublicRouteTable
  DefaultPublicRoute:
```

```
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1
  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2
  # Security Group
  ECSSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Security group for ECS tasks
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
        - IpProtocol: tcp
          FromPort: 443
          ToPort: 443
          CidrIp: 0.0.0.0/0
  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
```

```
  # CloudWatch Logs
  TaskLogGroup:
    Type: AWS::Logs::LogGroup
    DeletionPolicy: Retain
    UpdateReplacePolicy: Retain
    Properties:
      LogGroupName: !Ref LogGroupName
      RetentionInDays: 30
  # ECS Resources
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - Command:
            - >-
              /bin/sh -c "echo '<html> <head> <title>Amazon ECS Sample
              App</title> <style>body {margin-top: 40px; background-color:
              #333;} </style> </head><body> <div
              style=color:white;text-align:center> <h1>Amazon ECS Sample
              App</h1> <h2>Congratulations!</h2> <p>Your application is now
              running on a container in Amazon ECS.</p> </div></body></html>' >
              /usr/local/apache2/htdocs/index.html && httpd-foreground"s
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroupName
              awslogs-region: !Ref 'AWS::Region'
              awslogs-stream-prefix: ecs
          Name: sample-fargate-app
          PortMappings:
            - ContainerPort: 80
              HostPort: 80
```

```
                Protocol: tcp
      Cpu: !Ref TaskCpu
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      Family: !Ref TaskFamily
      Memory: !Ref TaskMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: LINUX
  ECSService:
    Type: AWS::ECS::Service
    DependsOn:
      - PublicSubnet1RouteTableAssociation
      - PublicSubnet2RouteTableAssociation
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ECSCluster
      DesiredCount: !Ref DesiredCount
      LaunchType: FARGATE
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
            - !Ref ECSSecurityGroup
          Subnets:
            - !Ref PublicSubnet1
            - !Ref PublicSubnet2
      TaskDefinition: !Ref ECSTaskDefinition
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref ECSTaskDefinition
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ECSService
  VpcId:
    Description: The ID of the VPC
    Value: !Ref VPC
  PublicSubnet1:
    Description: The ID of public subnet 1
```

```
    Value: !Ref PublicSubnet1
  PublicSubnet2:
    Description: The ID of public subnet 2
    Value: !Ref PublicSubnet2
  SecurityGroup:
    Description: The ID of the security group
    Value: !Ref ECSSecurityGroup
  ExecutionRoleArn:
    Description: The ARN of the task execution role
    Value: !GetAtt ECSTaskExecutionRole.Arn
```

3. Après avoir créé un fichier modèle, utilisez la commande suivante pour créer une pile. L'`--capabilities`indicateur est nécessaire pour créer un rôle d'exécution de tâche Amazon ECS tel que spécifié dans le modèle. Vous pouvez également spécifier l'`--parameters`indicateur pour personnaliser les paramètres du modèle.

```
aws cloudformation create-stack \
      --stack-name ecs-stack \
      --template-body file://ecs-tutorial-template.yaml \
      --capabilities CAPABILITY_IAM
```

Étape 2 : vérifier la création des ressources

Pour vous assurer que les ressources sont créées correctement, procédez comme suit. Vous pouvez également consulter la console Amazon ECS.

1. Exécutez la commande suivante pour répertorier toutes les définitions de tâches dans un Région AWS.

```
aws ecs list-task-definitions
```

La commande renvoie une liste de définitions de tâches Amazon Resource Name (ARN). L'ARN de la définition de tâche que vous avez créée à l'aide du modèle sera affiché au format suivant.

```
{
    "taskDefinitionArns": [
     .....
        "arn:aws:ecs:aws-region:111122223333:task-definition/task-definition-
cfn:1",
     .....
```

```
    ]
}
```

2. Exécutez la commande suivante pour répertorier tous les clusters d'un Région AWS.

```
aws ecs list-clusters
```

La commande renvoie une liste de clusters ARNs. L'ARN du cluster que vous avez créé à l'aide du modèle sera affiché au format suivant.

```
{
    "clusterArns": [
        .....
        "arn:aws:ecs:aws-region:111122223333:cluster/CFNCluster",
        .....
    ]
}
```

3. Exécutez la commande suivante pour répertorier tous les services du clusterCFNCluster.

```
aws ecs list-services \
      --cluster CFNCluster
```

La commande renvoie une liste de services ARNs. L'ARN du service que vous avez créé à l'aide du modèle sera affiché au format suivant.

```
{
    "serviceArns": [
        "arn:aws:ecs:aws-region:111122223333:service/CFNCluster/cfn-service"
    ]
}
```

Étape 3 : Nettoyer

Pour nettoyer les ressources que vous avez créées, exécutez la commande suivante.

```
aws cloudformation delete-stack \
      --stack-name ecs-stack
```

La `delete-stack` commande lance la suppression de la AWS CloudFormation pile créée dans ce didacticiel, en supprimant toutes les ressources de la pile. Pour vérifier la suppression, vous pouvez répéter la procédure dansÉtape 2 : vérifier la création des ressources. La liste ARNs des sorties n'inclura plus de définition de tâche appelée `task-definition-cfn` ni de cluster appelé`CFNCLuster`. L'`list-services`appel échouera.

Bonnes pratiques d'Amazon ECS

Vous pouvez utiliser l'une des pages suivantes pour découvrir les meilleures pratiques opérationnelles les plus importantes pour le réseau Amazon ECS.

Vue d'ensemble des meilleure s pratiques	En savoir plus
Connectez des applications à Internet	Connect les applications Amazon ECS à Internet
Recevez des connexions entrantes vers Amazon ECS depuis Internet.	Bonnes pratiques pour recevoir des connexions entrantes vers Amazon ECS depuis Internet
Connect Amazon ECS à d'autres AWS services au sein d'un VPC	Bonnes pratiques pour connecter Amazon ECS aux AWS services depuis votre VPC
Services réseau à travers Comptes AWS et VPCs	Bonnes pratiques pour la mise en réseau des services Amazon ECS entre Comptes AWS et VPCs
Résoudre les problèmes de réseau	AWS services de dépannage réseau Amazon ECS

Vous pouvez utiliser l'une des pages suivantes pour découvrir les meilleures pratiques opérationnelles les plus importantes pour Fargate sur Amazon ECS.

Vue d'ensemble des meilleures pratiques	En savoir plus	
Sécurité Fargate	Bonnes pratiques de sécurité de Fargate dans Amazon ECS	
Considérations relatives à la sécurité de Fargate	Considérations relatives à la sécurité de Fargate pour Amazon ECS	
Comportement d'extraction des images des conteneurs Linux sur le conteneur Fargate	Comportement d'extraction d'images de conteneurs Linux sur Fargate pour Amazon ECS	
Comportement d'extraction des images des conteneurs Windows sur le conteneur Fargate	Comportement d'extraction des images des conteneurs Windows sur Fargate pour Amazon ECS	
Retraite des tâches de Fargate	Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS	

Vous pouvez utiliser l'une des pages suivantes pour découvrir les meilleures pratiques opérationnelles les plus importantes en matière de définition des tâches.

Vue d'ensemble des meilleures pratiques	En savoir plus	
Images de conteneur	Bonnes pratiques pour les images de conteneurs Amazon ECS	
Taille de la tâche	Bonnes pratiques relatives à la taille des tâches Amazon ECS	

Vue d'ensemble des meilleures pratiques	En savoir plus	
Bonnes pratiques en matière de volumes	Options de stockage pour les tâches Amazon ECS	

Vous pouvez utiliser l'une des pages suivantes pour découvrir les meilleures pratiques opérationnelles les plus importantes en matière de clusters et de capacité.

Vue d'ensemble des meilleures pratiques	En savoir plus	
Sécurité Fargate	Bonnes pratiques de sécurité de Fargate dans Amazon ECS	
EC2 considérations relatives à la sécurité des instances de conteneur	Considérations relatives à la sécurité des instances de EC2 conteneur Amazon pour Amazon ECS	
Cluster Auto Scaling	Optimisation de la mise à l'échelle automatique du cluster Amazon ECS	
Fonctionnant à grande échelle	Exploiter Amazon ECS à grande échelle	
Mise à l'échelle automatique et gestion des capacités	Meilleures pratiques de mise à l'échelle automatique et de gestion des capacités d'Amazon ECS	

Vous pouvez utiliser l'une des pages suivantes pour découvrir les meilleures pratiques opérationnelles les plus importantes en matière de tâches et de services.

Vue d'ensemble des meilleures pratiques	En savoir plus
Optimisation du temps de lancement des tâches	Optimisez le temps de lancement des tâches Amazon ECS
Paramètres du service	Bonnes pratiques relatives aux paramètres de service Amazon ECS
Optimisation des paramètres de contrôle de santé de l'équilibreur de charge	Optimisation des paramètres de vérification de l'état de l'équilibreur de charge pour Amazon ECS
Optimisation des paramètres de vidange de la connexion de l'équilibreur de charge	Optimisez les paramètres de vidange des connexions de l'équilibreur de charge pour Amazon ECS
Optimisation de la mise à l'échelle automatique des services	Optimisation de la mise à l'échelle automatique du service Amazon ECS

Vous pouvez utiliser l'une des pages suivantes pour découvrir les meilleures pratiques opérationnelles les plus importantes en matière de sécurité.

Vue d'ensemble des meilleures pratiques	En savoir plus
Sécurité du réseau	Bonnes pratiques en matière de sécurité réseau pour Amazon ECS

Vue d'ensemble des meilleures pratiques	En savoir plus	
Sécurité des tâches et des conteneurs	Bonnes pratiques en matière de sécurité des tâches et des conteneurs Amazon ECS	

AWS Fargatepour Amazon ECS

AWS Fargateest une technologie que vous pouvez utiliser avec Amazon ECS pour exécuter des conteneurs sans avoir à gérer des serveurs ou des clusters d' EC2 instances Amazon. Avec AWS Fargate, vous n'avez plus besoin d'allouer, de configurer ou de mettre à l'échelle des clusters de machines virtuelles pour exécuter des conteneurs. Vous n'avez plus à choisir de types de serveurs, décider quand mettre à l'échelle vos clusters ni optimiser les packs de clusters.

Lorsque vous exécutez vos tâches et services avec le type de lancement Fargate, vous créez le package de votre application dans des conteneurs, spécifiez les exigences en termes de processeur et de mémoire, définissez la mise en réseau et les stratégies IAM, et vous lancez l'application. Chaque tâche Fargate possède sa propre limite d'isolement et ne partage pas le noyau sous-jacent, les ressources de processeur, les ressources de mémoire ou l'interface réseau Elastic avec une autre tâche. Vous configurez vos définitions de tâches pour Fargate en définissant le paramètre de définition de tâche `requiresCompatibilities` sur `FARGATE`. Pour de plus amples informations, veuillez consulter Types de lancement.

Fargate propose des versions de plateforme pour Amazon Linux 2 (version 1.3.0), le système d'exploitation Bottlerocket (version 1.4.0) et les éditions Microsoft Windows 2019 Server Full et Core. Sauf indication contraire, les informations de cette page s'appliquent à toutes les plateformes Fargate.

Cette rubrique décrit les différents composants des tâches et services Fargate et présente les considérations spéciales relatives à l'utilisation de Fargate avec Amazon ECS.

Pour de plus amples informations sur les Régions qui prennent en charge les conteneurs Linux sur Fargate, veuillez consulter the section called “Conteneurs Linux sur AWS Fargate”.

Pour de plus amples informations sur les Régions qui prennent en charge les conteneurs Windows sur Fargate, veuillez consulter the section called “Conteneurs Windows sur AWS Fargate”.

Procédures

Pour plus d'informations sur la façon de commencer à utiliser la console, voir :

- Découvrez comment créer une tâche Linux Amazon ECS pour le type de lancement Fargate
- Découvrez comment créer une tâche Windows Amazon ECS pour le type de lancement Fargate

Pour plus d'informations sur la façon de commencer à utiliser le AWS CLI, voir :

- Création d'une tâche Linux Amazon ECS pour le type de lancement Fargate avec AWS CLI
- Création d'une tâche Windows Amazon ECS pour le type de lancement Fargate avec AWS CLI

Fournisseurs de capacité

Les fournisseurs de capacité suivants sont disponibles :

- Fargate
- Fargate Spot : exécutez des tâches Amazon ECS tolérantes aux interruptions à un tarif réduit par rapport au prix. AWS Fargate Fargate Spot exécute les tâches sur la capacité de calcul de réserve. Lorsque vous aurez AWS besoin de retrouver votre capacité, vos tâches seront interrompues par un avertissement de deux minutes. Pour de plus amples informations, veuillez consulter Clusters Amazon ECS pour Fargate.

Définitions de tâche

Les tâches qui utilisent le type de lancement Fargate ne prennent pas en charge tous les paramètres de définition de tâche Amazon ECS disponibles. Certains paramètres ne sont pas du tout pris en charge, tandis que d'autres se comportent différemment pour les tâches Fargate. Pour de plus amples informations, veuillez consulter UC et mémoire au niveau de la tâche.

Versions de plateforme

AWS Les versions de la plate-forme Fargate sont utilisées pour faire référence à un environnement d'exécution spécifique pour l'infrastructure de tâches Fargate. Il s'agit d'une combinaison de la version Kernel et de la version d'exécution du conteneur. Vous sélectionnez une version de plateforme lorsque vous exécutez une tâche ou lorsque vous créez un service pour gérer un certain nombre de tâches identiques.

De nouvelles révisions des versions de plateforme sont publiées au fil de l'évolution de l'environnement d'exécution, par exemple si des mises à jour, de nouvelles fonctionnalités, des corrections de bugs ou des mises à jour de sécurité sont apportées au noyau ou au système d'exploitation. Une version de plateforme Fargate est mise à jour en effectuant une nouvelle révision de la version de plateforme. Chaque tâche s'exécute sur une révision de version de plateforme au cours de son cycle de vie. Si vous souhaitez utiliser la dernière version de plateforme, vous

devez démarrer une nouvelle tâche. Une nouvelle tâche exécutée sur Fargate s'exécute toujours sur la dernière version de plateforme, ce qui garantit que les tâches sont toujours lancées sur une infrastructure sécurisée et corrigée.

Si un problème de sécurité affectant une version de plate-forme existante est détecté, AWS crée une nouvelle révision corrigée de la version de plate-forme et met fin aux tâches exécutées sur la version vulnérable. Dans certains cas, vous pouvez être averti que la résiliation de vos tâches sur Fargate a été planifiée. Pour de plus amples informations, veuillez consulter Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS .

Pour de plus amples informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.

Équilibrage de charge des services

Vous pouvez configurer votre service Amazon ECS service sur AWS Fargate de manière à utiliser Elastic Load Balancing pour répartir uniformément le trafic entre les tâches de votre service.

Les services Amazon ECS AWS Fargate prennent en charge les types Application Load Balancer, Network Load Balancer et Load Balancer. Les équilibreurs de charge d'application sont utilisés pour acheminer le trafic HTTP/HTTPS (ou couche 7). Les équilibreurs de charge Network Load Balancer permettent d'acheminer le trafic TCP ou UDP (ou couche 4). Pour de plus amples informations, veuillez consulter Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS.

De même, lorsque vous créez des groupes cible pour ces services, vous devez choisir le type de cible `ip`, et non `instance`. Cela est dû au fait que les tâches qui utilisent le mode `awsvpc` réseau sont associées à une interface réseau élastique, et non à une EC2 instance Amazon. Pour de plus amples informations, veuillez consulter Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS.

L'utilisation d'un Network Load Balancer pour acheminer le trafic UDP vers votre Amazon ECS sur les tâches AWS Fargate n'est prise en charge que lors de l'utilisation de la plateforme version 1.4 ou ultérieure.

Métriques d'utilisation

Vous pouvez utiliser les statistiques CloudWatch d'utilisation pour obtenir une visibilité sur l'utilisation des ressources de votre compte. Utilisez ces indicateurs pour visualiser l'utilisation actuelle de vos services sur CloudWatch des graphiques et des tableaux de bord.

AWS Fargate les métriques d'utilisation correspondent aux quotas AWS de service. Vous pouvez configurer des alarmes qui vous alertent lorsque votre utilisation approche d'un quota de service. Pour plus d'informations sur les quotas AWS Fargate de service, les points de terminaison Amazon ECS et les quotas dans le... Référence générale d'Amazon Web Services

Pour plus d'informations sur les mesures AWS Fargate d'utilisation, consultez la section Mesures AWS Fargate d'utilisation.

Considérations relatives à la sécurité d'Amazon ECS concernant le moment d'utiliser le type de lancement Fargate

Nous recommandons aux clients qui recherchent une isolation solide pour leurs tâches d'utiliser Fargate. Fargate exécute chaque tâche dans un environnement de virtualisation matérielle. Cela garantit que ces charges de travail conteneurisées ne partagent pas les interfaces réseau, le stockage éphémère Fargate, le processeur ou la mémoire avec d'autres tâches. Pour plus d'informations, consultez la section Présentation de la sécurité de AWS Fargate.

Bonnes pratiques de sécurité de Fargate dans Amazon ECS

Nous vous recommandons de prendre en compte les bonnes pratiques ci-dessous lorsque vous utilisez AWS Fargate. Pour obtenir des conseils supplémentaires, consultez la section Présentation de la sécurité de AWS Fargate.

AWS KMS À utiliser pour chiffrer le stockage éphémère pour Fargate

Votre stockage éphémère doit être chiffré à l'aide de clés gérées par le client AWS KMS ou de vos propres clés. Pour les tâches hébergées sur Fargate à l'aide d'une `1.4.0` version de plateforme ou ultérieure, chaque tâche reçoit 20 GiB de stockage éphémère. Pour plus d'informations, consultez la section Clé gérée par le client (CMK). Vous pouvez augmenter la quantité totale de stockage éphémère, jusqu'à un maximum de 200 Gio, en spécifiant le paramètre `ephemeralStorage` dans votre définition de tâche. Pour les tâches lancées le 28 mai 2020 ou ultérieurement, le stockage éphémère est crypté à l'aide d'un algorithme de chiffrement AES-256 à l'aide d'une clé de chiffrement gérée par Fargate.

Pour plus d'informations, consultez la section Options de stockage pour les tâches Amazon ECS.

Exemple : lancement d'une tâche sur la plateforme Fargate version 1.4.0 avec chiffrement de stockage éphémère

La commande suivante lancera une tâche sur la version 1.4 de la plateforme Fargate. Comme cette tâche est lancée dans le cadre du cluster, elle utilise les 20 GiB de stockage éphémère qui sont automatiquement chiffrés.

```
aws ecs run-task --cluster clustername \
  --task-definition taskdefinition:version \
  --count 1
  --launch-type "FARGATE" \
  --platform-version 1.4.0 \
  --network-configuration
 "awsvpcConfiguration={subnets=[subnetid],securityGroups=[securitygroupid]}" \
  --region region
```

Fonctionnalité SYS_PTRACE pour le suivi des appels système du noyau avec Fargate

La configuration par défaut des fonctionnalités Linux ajoutées ou supprimées de votre conteneur est fournie par Docker.

Les tâches lancées sur Fargate ne prennent en charge que l'ajout de la capacité du noyau `SYS_PTRACE`.

La vidéo suivante montre comment utiliser cette fonctionnalité dans le cadre du projet Sysdig Falco.

ContainersFromTheCouch - Résolution des problèmes liés à votre tâche Fargate à l'aide de la fonctionnalité SYS_PTRACE

Le code décrit dans la vidéo précédente se trouve GitHub ici.

Utilisez Amazon GuardDuty avec Fargate Runtime Monitoring

Amazon GuardDuty est un service de détection des menaces qui aide à protéger vos comptes, vos conteneurs, vos charges de travail et les données de votre AWS environnement. À l'aide de modèles d'apprentissage automatique (ML) et de capacités de détection des anomalies et des menaces, vous surveillez GuardDuty en permanence les différentes sources de journaux et l'activité d'exécution afin d'identifier et de hiérarchiser les risques de sécurité potentiels et les activités malveillantes dans votre environnement.

La surveillance du temps d'exécution GuardDuty protège les charges de travail exécutées sur Fargate en AWS surveillant en permanence l'activité des journaux et du réseau afin d'identifier les comportements malveillants ou non autorisés. Runtime Monitoring utilise un agent de GuardDuty

sécurité léger et entièrement géré qui analyse le comportement sur l'hôte, tel que l'accès aux fichiers, l'exécution des processus et les connexions réseau. Cela couvre des problèmes tels que l'augmentation des privilèges, l'utilisation d'informations d'identification divulguées ou la communication avec des adresses IP ou des domaines malveillants, ainsi que la présence de logiciels malveillants sur vos EC2 instances Amazon et vos charges de travail de conteneur. Pour plus d'informations, consultez la section Surveillance du temps GuardDuty d'exécution dans le guide de GuardDuty l'utilisateur.

Considérations relatives à la sécurité de Fargate pour Amazon ECS

Chaque tâche dispose d'une capacité d'infrastructure dédiée, car Fargate exécute chaque charge de travail dans un environnement virtuel isolé. Les charges de travail exécutées sur Fargate ne partagent pas les interfaces réseau, le stockage éphémère, le processeur ou la mémoire avec d'autres tâches. Vous pouvez exécuter plusieurs conteneurs au sein d'une même tâche, notamment des conteneurs d'applications et des conteneurs sidecar, ou simplement des sidecars. Un sidecar est un conteneur qui s'exécute parallèlement à un conteneur d'application dans une tâche Amazon ECS. Alors que le conteneur d'applications exécute le code de base de l'application, les processus exécutés dans des sidecars peuvent enrichir l'application. Les sidecars vous aident à séparer les fonctions de l'application dans des conteneurs dédiés, ce qui facilite la mise à jour de certaines parties de votre application.

Les conteneurs qui font partie de la même tâche partagent des ressources pour le type de lancement Fargate, car ils s'exécuteront toujours sur le même hôte et partageront les ressources de calcul. Ces conteneurs partagent également le stockage éphémère fourni par Fargate. Les conteneurs Linux d'une tâche partagent des espaces de noms réseau, notamment l'adresse IP et les ports réseau. À l'intérieur d'une tâche, les conteneurs appartenant à la tâche peuvent communiquer entre eux via l'hôte local.

L'environnement d'exécution de Fargate vous empêche d'utiliser certaines fonctionnalités du contrôleur prises en charge sur les instances. EC2 Tenez compte des éléments suivants lorsque vous concevez des charges de travail qui s'exécutent sur Fargate :

- Aucun conteneur ou accès privilégié : les fonctionnalités telles que les conteneurs ou l'accès privilégié ne sont actuellement pas disponibles sur Fargate. Cela affectera les cas d'utilisation tels que l'exécution de Docker dans Docker.
- Accès limité aux fonctionnalités de Linux : l'environnement dans lequel les conteneurs s'exécutent sur Fargate est verrouillé. Les fonctionnalités Linux supplémentaires, telles que CAP_SYS_ADMIN et CAP_NET_ADMIN, sont limitées afin d'empêcher une escalade des privilèges. Fargate prend

en charge l'ajout de la fonctionnalité Linux CAP_SYS_PTRACE aux tâches afin de permettre aux outils d'observabilité et de sécurité déployés dans le cadre de la tâche de surveiller l'application conteneurisée.

- Aucun accès à l'hôte sous-jacent : ni les clients ni AWS les opérateurs ne peuvent se connecter à un hôte exécutant les charges de travail des clients. Vous pouvez utiliser ECS Exec pour exécuter des commandes dans ou obtenir un shell vers un conteneur s'exécutant sur Fargate. Vous pouvez utiliser ECS Exec pour collecter des informations de diagnostic pour le débogage. Fargate empêche également les conteneurs d'accéder aux ressources de l'hôte sous-jacent, telles que le système de fichiers, les périphériques, la mise en réseau et l'environnement d'exécution du conteneur.
- Mise en réseau : vous pouvez utiliser les groupes de sécurité et ACLs le réseau pour contrôler le trafic entrant et sortant. Les tâches Fargate reçoivent une adresse IP depuis le sous-réseau configuré de votre VPC.

Versions de la plateforme Fargate pour Amazon ECS

AWS Les versions de la plate-forme Fargate sont utilisées pour faire référence à un environnement d'exécution spécifique pour l'infrastructure de tâches Fargate. Il s'agit d'une combinaison de la version Kernel et de la version d'exécution du conteneur. Vous sélectionnez une version de plateforme lorsque vous exécutez une tâche ou lorsque vous créez un service pour gérer un certain nombre de tâches identiques.

De nouvelles révisions des versions de plateforme sont publiées au fil de l'évolution de l'environnement d'exécution, par exemple si des mises à jour, de nouvelles fonctionnalités, des corrections de bugs ou des mises à jour de sécurité sont apportées au noyau ou au système d'exploitation. Une version de plateforme Fargate est mise à jour en effectuant une nouvelle révision de la version de plateforme. Chaque tâche s'exécute sur une révision de version de plateforme au cours de son cycle de vie. Si vous souhaitez utiliser la dernière version de plateforme, vous devez démarrer une nouvelle tâche. Une nouvelle tâche exécutée sur Fargate s'exécute toujours sur la dernière version de plateforme, ce qui garantit que les tâches sont toujours lancées sur une infrastructure sécurisée et corrigée.

Si un problème de sécurité affectant une version de plate-forme existante est détecté, AWS crée une nouvelle révision corrigée de la version de plate-forme et met fin aux tâches exécutées sur la version vulnérable. Dans certains cas, vous pouvez être averti que la résiliation de vos tâches sur Fargate a été planifiée. Pour de plus amples informations, veuillez consulter Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS .

Vous spécifiez la version de la plateforme lorsque vous exécutez une tâche ou que vous déployez un service.

Tenez compte des éléments suivants lorsque vous spécifiez une version de plateforme :

- Vous pouvez spécifier un numéro de version spécifique, par exemple`1.4.0`, ou`LATEST`.

  La dernière version de la plateforme Linux est`1.4.0`.

  La dernière version de la plate-forme Windows est`1.0.0`.
- Si vous souhaitez mettre à jour la version de plateforme d'un service, créez un déploiement. Par exemple, supposons que vous ayez un service qui exécute des tâches sur la version de plateforme Linux `1.3.0`. Pour modifier le service afin d'exécuter des tâches sur la version de la plate-forme Linux`1.4.0`, vous devez mettre à jour votre service et spécifier une nouvelle version de plate-forme. Vos tâches sont redéployées avec la dernière version de plateforme et la dernière révision de la version de plateforme. Pour de plus amples informations sur les déploiements, veuillez consulter Services Amazon ECS.
- Si votre service est mis à l'échelle sans que la version de plateforme soit mise à jour, ces tâches reçoivent la version de plateforme spécifiée dans le déploiement actuel de ce service. Par exemple, supposons que vous ayez un service qui exécute des tâches sur la version de plateforme Linux `1.3.0`. Si vous augmentez le nombre souhaité de services, le planificateur de services lance les nouvelles tâches en utilisant la dernière version de plateforme, révision de la version de plateforme `1.3.0`.
- Les nouvelles tâches sont toujours exécutées sur la dernière révision d'une version de la plateforme. Cela garantit que les tâches sont toujours effectuées sur une infrastructure sécurisée et corrigée.
- Les numéros de version de plateforme pour les conteneurs Linux et les conteneurs Windows sur Fargate sont indépendants. Par exemple, le comportement, les fonctionnalités et les logiciels utilisés dans la version de plateforme `1.0.0` pour les conteneurs Windows sur Fargate ne sont pas comparables à ceux de la version de plateforme `1.0.0` pour les conteneurs Linux sur Fargate.
- Ce qui suit s'applique aux versions de la plateforme Fargate Windows.

  Les images de conteneur Microsoft Windows Server doivent être créées à partir d'une version spécifique de Windows Server. Vous devez sélectionner la même version de Windows Server dans la `platformFamily` lorsque vous exécutez une tâche ou créez un service correspondant à l'image du conteneur Windows Server. En outre, vous pouvez fournir une correspondance

`operatingSystemFamily` dans la définition de tâche afin d'éviter que les tâches ne soient exécutées sur la mauvaise version de Windows. Pour plus d'informations, veuillez consulter Correspondance de la version d'hôte de conteneur avec les versions des images de conteneur sur le site Web de Microsoft Learn.

Migration vers la version 1.4.0 de la plateforme Linux sur Amazon ECS

Tenez compte des éléments suivants lors de la migration de vos tâches Amazon ECS sur Fargate à partir de la version `1.0.0`, `1.1.0`, `1.2.0` ou `1.3.0` de la plateforme vers la version `1.4.0`. Il est recommandé de vérifier que votre tâche fonctionne correctement sur la version de la plateforme `1.4.0` avant de migrer les tâches.

- Le comportement du trafic réseau entrant et sortant entre les tâches a été mis à jour. À partir de la version 1.4.0 de la plateforme, toutes les tâches Amazon ECS sur Fargate reçoivent une seule interface réseau élastique (appelée ENI de tâches) et tout le trafic réseau passe par cette ENI au sein de votre VPC. Le trafic est visible par le biais de vos journaux de flux VPC. Pour de plus amples informations, veuillez consulter Options de mise en réseau des tâches Amazon ECS pour le type de lancement Fargate.
- Si vous utilisez des points de terminaison VPC d'interface, tenez compte des points suivants.
  - Pour les images de conteneur hébergées avec Amazon ECR, vous avez besoin des points de terminaison suivants. Pour plus d'informations, consultez Points de terminaison d'un VPC d'interface Amazon ECR (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.
    - com.amazonaws. ***region***.ecr.dkr Point de terminaison VPC Amazon ECR
    - com.amazonaws. ***region***.ecr.api Point de terminaison VPC Amazon ECR
    - Point de terminaison de la passerelle Amazon S3
  - Lorsque votre définition de tâche fait référence aux secrets de Secrets Manager pour récupérer des données sensibles pour vos conteneurs, vous devez créer les points de terminaison VPC d'interface pour Secrets Manager. Pour de plus amples informations, veuillez consulter Utilisation de Secrets Manager avec des points de terminaison de VPC dans le Guide de l'utilisateur AWS Secrets Manager .
  - Lorsque votre définition de tâche fait référence aux paramètres du magasin de paramètres de Systems Manager pour récupérer des données sensibles pour vos conteneurs, vous devez créer les points de terminaison VPC de l'interface pour Systems Manager. Pour plus d'informations, consultez la section Améliorer la sécurité des EC2 instances en utilisant les

points de terminaison VPC pour Systems Manager dans le guide de l'AWS Systems Manager utilisateur.

- Le groupe de sécurité pour l'Elastic Network Interface (ENI) associé à votre tâche a besoin des règles du groupe de sécurité pour autoriser le trafic entre la tâche et les points de terminaison VPC.

Journal des modifications de version de la plateforme Fargate Linux

Les versions de plateforme Linux suivantes sont proposées. Pour obtenir des informations sur l'obsolescence des versions de plateforme, consultez AWS Version obsolète de la plateforme Fargate Linux.

1.4.0

Voici le journal des modifications de la version `1.4.0` de la plateforme.

- Depuis le 5 novembre 2020, toute nouvelle tâche Amazon ECS lancée sur Fargate en utilisant la version `1.4.0` de la plateforme peut utiliser les fonctions suivantes :
  - Lorsque vous utilisez Secrets Manager pour stocker des données sensibles, vous pouvez injecter une clé JSON spécifique ou une version spécifique d'un secret en tant que variable d'environnement ou dans une configuration de journal. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.
  - Spécifiez les variables d'environnement en bloc à l'aide du paramètre de définition de conteneur `environmentFiles`. Pour de plus amples informations, veuillez consulter Transmettre une variable d'environnement individuelle à un conteneur Amazon ECS.
  - Les tâches exécutées dans un VPC et un sous-réseau activé pour IPv6 se verront attribuer à la fois une IPv4 adresse privée et une adresse. IPv6 Pour plus d'informations, consultez les options de mise en réseau des tâches Amazon ECS pour le type de lancement Fargate.
  - Le point de terminaison de métadonnées de tâche version 4 fournit des métadonnées supplémentaires sur votre tâche et conteneur, y compris le type de lancement de tâche, l'Amazon Resource Name (ARN) du conteneur, ainsi que le pilote de journal et les options de pilote de journal utilisés. Lors de l'interrogation du point de terminaison `/stats`, vous recevez également des statistiques de débit réseau pour vos conteneurs. Pour plus d'informations, consultez la version 4 du point de terminaison des métadonnées des tâches.
- Depuis le 30 juillet 2020, toute nouvelle tâche Amazon ECS lancée sur Fargate en utilisant la version `1.4.0` de la plateforme peut acheminer le trafic UDP à l'aide d'un Network Load Balancer

vers Amazon ECS sur les tâches Fargate. Pour de plus amples informations, veuillez consulter Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS.

- À compter du 28 mai 2020, toute nouvelle tâche Amazon ECS lancée sur Fargate à l'aide de la `1.4.0` version de la plateforme verra son stockage éphémère chiffré à l'aide d'un algorithme de chiffrement AES-256 utilisant une clé de chiffrement propriétaire. AWS Pour plus d'informations, consultez Stockage éphémère des tâches Fargate pour Amazon ECS et Options de stockage pour les tâches Amazon ECS.
- Ajout de la prise en charge de l'utilisation des volumes de système de fichiers Amazon EFS pour le stockage des tâches permanentes. Pour de plus amples informations, veuillez consulter Utiliser les volumes Amazon EFS avec Amazon ECS.
- Le stockage des tâches éphémères est désormais d'au moins 20 Go pour chaque tâche. Pour de plus amples informations, veuillez consulter Stockage éphémère des tâches Fargate pour Amazon ECS.
- Le comportement du trafic réseau entrant et sortant entre les tâches a été mis à jour. À partir de la version 1.4.0 de la plateforme, toutes les tâches Fargate reçoivent une interface réseau Elastic unique (appelée ENI de tâche), et tout le trafic réseau passe par cette ENI au sein du VPC et vous est accessible via les journaux de vos flux VPC. Pour plus d'informations sur la mise en réseau pour le type de EC2 lancement Amazon, consultez les options de mise en réseau des tâches Amazon ECS pour le type de EC2 lancement. Pour plus d'informations sur la mise en réseau pour le type de lancement Fargate, consultez. Options de mise en réseau des tâches Amazon ECS pour le type de lancement Fargate
- Tâche : ENIs ajouter un support pour les cadres jumbo. Les interfaces réseau sont configurées avec une unité de transmission maximale (MTU), qui correspond à la taille de la charge utile la plus élevée possible dans une seule trame. Plus la MTU est importante, plus la charge utile de l'application peut s'intégrer dans une seule trame. Cela réduit les frais généraux par trame et augmente l'efficacité. La prise en charge des trames jumbo limite la surcharge lorsque le chemin réseau entre votre tâche et la destination prend en charge les trames jumbo, telles que tout le trafic qui reste dans votre VPC.
- CloudWatch Container Insights inclura des indicateurs de performance réseau pour les tâches Fargate. Pour de plus amples informations, veuillez consulter Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée.
- Ajout de la prise en charge du point de terminaison de métadonnées de tâche version 4, qui fournit des informations supplémentaires pour vos tâches Fargate, y compris les statistiques réseau pour la tâche et la zone de disponibilité dans laquelle la tâche s'exécute. Pour plus d'informations,

consultez Point de terminaison des métadonnées des tâches Amazon ECS, version 4 et Point de terminaison de métadonnées de tâches Amazon ECS version 4 pour les tâches sur Fargate.

- Ajout de la prise en charge du paramètre `SYS_PTRACE` Linux dans les définitions de conteneur. Pour de plus amples informations, veuillez consulter Paramètres Linux.
- L'agent de conteneur Fargate remplace l'agent de conteneur Amazon ECS pour toutes les tâches Fargate. Généralement, cette modification ne devrait pas avoir d'effet sur la façon dont vos tâches s'exécutent.
- L'environnement d'exécution du conteneur utilise maintenant Containerd au lieu de Docker. Dans la plupart des cas, cette modification ne devrait pas avoir d'effet sur la façon dont vos tâches s'exécutent. Vous remarquerez que certains messages d'erreur provenant de l'environnement d'exécution du conteneur mentionnent de moins en moins Docker et de en plus des erreurs générales. Pour plus d'informations, consultez les messages d'erreur relatifs à l'arrêt de la tâche Amazon ECS.
- Basée sur Amazon Linux 2.

1.3.0

Voici le journal des modifications de la version `1.3.0` de la plateforme.

- Depuis le 30 septembre 2019, toute nouvelle tâche Fargate lancée prend en charge le pilote de journal `awsfirelens`. Configurez le FireLens pour qu'Amazon ECS utilise les paramètres de définition des tâches pour acheminer les journaux vers un AWS service ou une destination du réseau de AWS partenaires (APN) à des fins de stockage et d'analyse des journaux. Pour de plus amples informations, veuillez consulter Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.
- Ajout du recyclage de tâche pour les tâches Fargate, qui est le processus d'actualisation des tâches qui font partie d'un service Amazon ECS service. Pour plus d'informations, consultez la section Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS.
- Depuis le 27 mars 2019, toute nouvelle tâche Fargate lancée peut utiliser des paramètres de définition de tâche supplémentaires qui vous permettent de définir une configuration proxy, des dépendances de démarrage et d'arrêt de conteneurs, ainsi qu'une valeur de temporisation d'arrêt et de démarrage par conteneur. Pour plus d’informations, consultez Configuration du proxy, Dépendances du conteneur et Temporisations de conteneurs.
- À compter du 2 avril 2019, toute nouvelle tâche Fargate lancée prend en charge l'injection de données sensibles dans vos conteneurs en stockant vos données sensibles dans des secrets

AWS Secrets Manager AWS Systems Manager ou dans des paramètres Parameter Store, puis en les référençant dans la définition de votre conteneur. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.

- Depuis le 1er mai 2019, toute nouvelle tâche Fargate lancée prend en charge le référencement des données sensibles dans la configuration de journal d'un conteneur à l'aide du paramètre de définition de conteneur `secretOptions`. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.
- Depuis le 1er mai 2019, toute nouvelle tâche Fargate lancée prend en charge le pilote de journal `splunk` en plus du pilote de journal `awslogs`. Pour de plus amples informations, veuillez consulter Stockage et journalisation.
- À compter du 9 juillet 2019, toutes les nouvelles tâches Fargate lancées CloudWatch seront prises en charge par Container Insights. Pour de plus amples informations, veuillez consulter Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée.
- Depuis le 3 décembre 2019, le fournisseur de capacité Fargate Spot est pris en charge. Pour de plus amples informations, veuillez consulter Clusters Amazon ECS pour Fargate.
- Basée sur Amazon Linux 2.

AWS Version obsolète de la plateforme Fargate Linux

Cette page répertorie les versions de la plate-forme Linux que AWS Fargate a déconseillées ou dont l'obsolescence a été planifiée. Ces versions de plateforme restent disponibles jusqu'à la date d'obsolescence publiée.

Une date de mise à jour forcée est fournie pour chaque version de plateforme dont l'obsolescence est programmée. À la date de mise à jour forcée, tout service utilisant la version de plateforme `LATEST` qui est indiquée comme une version dont l'obsolescence est programmée sera mise à jour à l'aide de l'option Forcer le nouveau déploiement. Lorsque le service est mis à jour à l'aide de l'option Forcer le nouveau déploiement, toutes les tâches exécutées sur une version de plateforme dont l'obsolescence est programmée sont arrêtées et de nouvelles tâches sont lancées à l'aide de la version de plateforme que la balise `LATEST` indique à ce moment. Les tâches ou services autonomes avec un jeu de version de plateforme explicite ne sont pas affectés par la date de mise à jour forcée.

Nous vous recommandons de mettre à jour les tâches autonomes de vos services afin d'utiliser la version la plus récente de la plateforme. Pour plus d'informations sur la migration vers la version la plus récente de la plateforme, consultez Migration vers la version 1.4.0 de la plateforme Linux sur Amazon ECS.

Une fois qu'une version de plateforme atteint sa date d'obsolescence, la version de plateforme ne sera plus disponible pour de nouvelles tâches ou services. Toutes les tâches ou services autonomes qui utilisent explicitement une version de plateforme obsolète continueront d'utiliser cette version de plateforme jusqu'à ce que les tâches soient arrêtées. Après la date d'obsolescence, une version de plateforme obsolète ne recevra plus de mises à jour de sécurité ou de corrections de bugs.

Version de plateforme	Date de mise à jour forcée	Date d'obsolescence
1.0.0	26 octobre 2020	14 décembre 2020
1.1.0	26 octobre 2020	14 décembre 2020
1.2.0	26 octobre 2020	14 décembre 2020

Pour obtenir des informations sur les versions de plateforme actuelles, consultez Versions de la plateforme Fargate pour Amazon ECS.

Journal des modifications des versions obsolètes de Fargate Linux

1.2.0

Voici le journal des modifications de la version `1.2.0` de la plateforme.

> **Note**
>
> La version de plateforme `1.2.0` n'est plus disponible. Pour obtenir des informations sur l'obsolescence des versions de plateforme, consultez AWS Version obsolète de la plateforme Fargate Linux.

- Ajout du support pour l'authentification du registre privé à l'aide de AWS Secrets Manager. Pour de plus amples informations, veuillez consulter Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.

1.1.0

Voici le journal des modifications de la version `1.1.0` de la plateforme.

> **Note**
>
> La version de plateforme `1.1.0` n'est plus disponible. Pour obtenir des informations sur l'obsolescence des versions de plateforme, veuillez consulter AWS Version obsolète de la plateforme Fargate Linux.

- Ajout de la prise en charge du point de terminaison des métadonnées de tâches Amazon ECS. Pour de plus amples informations, veuillez consulter Métadonnées des tâches Amazon ECS disponibles pour les tâches sur Fargate.
- Ajout de la prise en charge des surveillances de l'état Docker dans les définitions de conteneur. Pour de plus amples informations, veuillez consulter Surveillance de l'état.
- Ajout de la prise en charge de la découverte de service Amazon ECS service. Pour de plus amples informations, veuillez consulter Utilisez la découverte des services pour connecter les services Amazon ECS aux noms DNS.

1.0.0

Voici le journal des modifications de la version `1.0.0` de la plateforme.

> **Note**
>
> La version de plateforme `1.0.0` n'est plus disponible. Pour obtenir des informations sur l'obsolescence des versions de plateforme, veuillez consulter AWS Version obsolète de la plateforme Fargate Linux.

- Basée sur Amazon Linux 2017.09.
- Première version.

Journal des modifications de version de la plateforme Windows Fargate

Les versions de plateforme pour les conteneurs Windows suivantes sont proposées.

1.0.0

Voici le journal des modifications de la version `1.0.0` de la plateforme.

- Version initiale pour la prise en charge des systèmes d'exploitation Microsoft Windows Server suivants :
  - Windows Server 2019 Full
  - Windows Server 2019 Core
  - Windows Server 2022 Full
  - Windows Server 2022 Core

Considérations relatives aux conteneurs Windows sur Fargate pour Amazon ECS

Voici les différences et les considérations à prendre en compte lorsque vous exécutez des conteneurs Windows sur AWS Fargate.

Si vous devez exécuter des tâches sur des conteneurs Linux et Windows, vous devez créer des définitions de tâches distinctes pour chaque système d'exploitation.

AWS gère la gestion des licences du système d'exploitation, vous n'avez donc pas besoin de licences Microsoft Windows Server supplémentaires.

Les conteneurs Windows sur AWS Fargate sont compatibles avec les systèmes d'exploitation suivants :

- Windows Server 2019 Full
- Windows Server 2019 Core
- Windows Server 2022 Full
- Windows Server 2022 Core

Les conteneurs Windows de AWS Fargate prennent en charge le pilote awslogs. Pour de plus amples informations, veuillez consulter the section called “Envoyer les journaux à CloudWatch ”.

Les fonctionnalités suivantes ne sont pas prises en charge sur les conteneurs Windows sur Fargate :

- Amazon FSx
- Jonction ENI
- g MSAs pour Windows Containers
- Les intégrations du service App Mesh et du proxy pour les tâches

- L'intégration du routeur journal Firelens pour les tâches
- Les volumes EFS
- Volumes EBS
- Les paramètres de définition de tâche suivants :
  - `maxSwap`
  - `swappiness`
  - `environmentFiles`
- Le fournisseur de capacités Fargate Spot
- Les volumes d'images

  L'option Dockerfile `volume` est ignorée. Au lieu de cela, utilisez des montages liés dans votre définition de tâche. Pour de plus amples informations, veuillez consulter Utiliser des montages par liaison avec Amazon ECS.
- Les paramètres du processeur et de la mémoire au niveau des tâches sont ignorés pour les conteneurs Windows. Nous vous recommandons de spécifier des ressources de niveau conteneur pour les conteneurs Windows.
- mémoire pour la tâche
- MermoryReservation sur les conteneurs
- Politiques de redémarrage relatives aux conteneurs

Stockage éphémère des tâches Fargate pour Amazon ECS

Une fois provisionnée, chaque tâche Amazon ECS hébergée sur des conteneurs Linux AWS Fargate reçoit le stockage éphémère suivant pour les montages liés. Ils peuvent être montés et partagés entre les conteneurs en utilisant les paramètres `volumes`, `mountPoints` et `volumesFrom` dans la définition de tâche. Cela n'est pas pris en charge pour les conteneurs Windows sur AWS Fargate.

Versions de la plateforme de conteneurs Linux Fargate

Version 1.4.0 ou ultérieure

Par défaut, les tâches Amazon ECS sur Fargate utilisant la version `1.4.0` ou ultérieure de la plateforme reçoivent un minimum de 20 Gio de stockage éphémère. La quantité totale de stockage éphémère peut être augmentée, jusqu'à un maximum de 200 Gio. Vous pouvez effectuer cette opération en spécifiant le paramètre `ephemeralStorage` dans votre définition de tâche.

Les images de conteneur extraites, compressées et non compressées de la tâche sont stockées sur le stockage éphémère. Pour déterminer la quantité totale de stockage éphémère que votre tâche doit utiliser, vous devez soustraire la quantité de stockage utilisée par votre image de conteneur de la quantité totale de stockage éphémère alloué à votre tâche.

Pour les tâches utilisant la version `1.4.0` ou ultérieure de la plateforme qui sont lancées le 28 mai 2020 ou plus tard, le stockage éphémère est chiffré à l'aide d'un algorithme de chiffrement AES-256. Cet algorithme utilise une AWS clé de chiffrement propre, ou vous pouvez créer votre propre clé gérée par le client. Pour plus d'informations, consultez la section Clés gérées par le client pour le AWS Fargate stockage éphémère.

Pour les tâches utilisant la version `1.4.0` ou ultérieure de la plateforme qui sont lancées le 18 novembre 2022 ou plus tard, l'utilisation du stockage éphémère est signalée sur le point de terminaison des métadonnées de la tâche. Dans le cadre de vos tâches, vos applications peuvent interroger le point de terminaison des métadonnées des tâches, version `4`, pour connaître la taille réservée au stockage éphémère et la quantité utilisée.

En outre, la taille réservée au stockage éphémère et la quantité utilisée sont envoyées à Amazon CloudWatch Container Insights si vous activez Container Insights.

> **Note**
>
> Fargate réserve de l'espace sur le disque. Il n'est utilisé que par Fargate. Vous n'êtes pas facturé pour cela. Cela n'apparaît pas dans ces statistiques. Toutefois, vous pouvez voir ce stockage supplémentaire dans d'autres outils tels que `df`.

Version 1.3.0 ou antérieure

Pour les tâches Amazon ECS sur Fargate utilisant la version `1.3.0` ou antérieure de la plateforme, chaque tâche reçoit le magasin éphémère suivant.

- 10 Go de stockage de couche Docker

> **Note**
>
> Ce volume inclut les artefacts d'image de conteneur compressée et non compressée.

- 4 Go supplémentaires pour les montages de volume. Ils peuvent être montés et partagés entre les conteneurs en utilisant les paramètres `volumes`, `mountPoints` et `volumesFrom` dans la définition de tâche.

Versions de la plateforme de conteneur Windows Fargate

Version 1.0.0 ou ultérieure

Par défaut, les tâches Amazon ECS sur Fargate utilisant la version `1.0.0` ou ultérieure de la plateforme reçoivent un minimum de 20 Gio de stockage éphémère. La quantité totale de stockage éphémère peut être augmentée, jusqu'à un maximum de 200 Gio. Vous pouvez effectuer cette opération en spécifiant le paramètre `ephemeralStorage` dans votre définition de tâche.

Les images de conteneur extraites, compressées et non compressées de la tâche sont stockées sur le stockage éphémère. Pour déterminer la quantité totale de stockage éphémère que votre tâche doit utiliser, vous devez soustraire la quantité de stockage utilisée par votre image de conteneur de la quantité totale de stockage éphémère alloué à votre tâche.

Pour de plus amples informations, veuillez consulter Utiliser des montages par liaison avec Amazon ECS.

Clés gérées par le client pour le stockage AWS Fargate éphémère pour Amazon ECS

AWS Fargate prend en charge les clés gérées par le client pour chiffrer les données relatives aux tâches Amazon ECS stockées dans un stockage éphémère afin d'aider les clients sensibles aux réglementations à respecter leurs politiques de sécurité internes. Les clients continuent de bénéficier des avantages de Fargate sans serveur, tout en offrant aux auditeurs de conformité une meilleure visibilité sur le chiffrement du stockage autogéré. Bien que Fargate utilise le chiffrement du stockage éphémère géré par Fargate par défaut, les clients peuvent également utiliser leurs propres clés autogérées pour chiffrer des données sensibles telles que des informations financières ou liées à la santé.

Vous pouvez importer vos propres clés AWS KMS ou créer des clés dans AWS KMS. Ces clés autogérées sont stockées AWS KMS et exécutent des actions de AWS KMS cycle de vie standard telles que la rotation, la désactivation et la suppression. Vous pouvez vérifier l'accès aux clés et leur utilisation dans CloudTrail les journaux.

Par défaut, la clé KMS prend en charge 50 000 subventions par clé. Fargate utilise une AWS KMS seule subvention par tâche clé gérée par le client, ce qui lui permet de prendre en charge jusqu'à 50 000 tâches simultanées pour une clé. Si vous souhaitez augmenter ce nombre, vous pouvez demander une augmentation de limite, qui est approuvée sur une case-by-case base régulière.

Fargate ne facture aucun supplément pour l'utilisation des clés gérées par le client. Le prix standard ne vous est facturé que pour l'utilisation des AWS KMS clés pour le stockage et les demandes d'API.

Rubriques

- Création d'une clé de chiffrement pour le stockage éphémère Fargate pour Amazon ECS
- Gestion des AWS KMS clés pour le stockage éphémère Fargate pour Amazon ECS

Création d'une clé de chiffrement pour le stockage éphémère Fargate pour Amazon ECS

Créez une clé gérée par le client pour chiffrer les données stockées sur le stockage éphémère Fargate.

> **Note**
>
> Le chiffrement du stockage éphémère Fargate avec des clés gérées par le client n'est pas disponible pour les clusters de tâches Windows.
> Le chiffrement du stockage éphémère Fargate avec des clés gérées par le client n'est pas disponible avant. `platformVersions` `1.4.0`
> Fargate réserve de l'espace sur un espace de stockage éphémère qui n'est utilisé que par Fargate, et cet espace ne vous est pas facturé. L'allocation peut être différente de celle des tâches clés non gérées par le client, mais l'espace total reste le même. Vous pouvez visualiser cette modification dans des outils tels que`df`.
> Les clés multirégionales ne sont pas prises en charge pour le stockage éphémère Fargate.
> Les alias de clé KMS ne sont pas pris en charge pour le stockage éphémère Fargate.

Pour créer une clé gérée par le client (CMK) afin de chiffrer le stockage éphémère pour Fargate in, procédez comme suit. AWS KMS

1. Accédez au fichier https://console.aws.amazon.com/km.

2. Suivez les instructions relatives à la création de clés dans le guide du AWS Key Management Service développeur.
3. Lorsque vous créez votre AWS KMS clé, assurez-vous de fournir les autorisations d'opération AWS KMS pertinentes pour le service Fargate dans les politiques clés. Les opérations d'API suivantes doivent être autorisées dans la politique pour utiliser votre clé gérée par le client avec les ressources de votre cluster Amazon ECS.

   - `kms:GenerateDataKeyWithoutPlainText`- Appelez `GenerateDataKeyWithoutPlainText` pour générer une clé de données cryptée à partir de la AWS KMS clé fournie.
   - `kms:CreateGrant`- Ajoute une subvention à une clé gérée par le client. Accorde un accès de contrôle à une AWS KMS clé spécifiée, ce qui permet d'autoriser les opérations requises par Amazon ECS Fargate. Pour plus d'informations sur l'utilisation des subventions, consultez le guide du AWS Key Management Service développeur. Cela permet à Amazon ECS Fargate d'effectuer les opérations suivantes :
     - Appelez `Decrypt` pour AWS KMS obtenir la clé de chiffrement permettant de déchiffrer les données de stockage éphémères.
     - Configurez un directeur partant à la retraite pour permettre au service de`RetireGrant`.
   - `kms:DescribeKey`- Fournit les détails de la clé gérée par le client pour permettre à Amazon ECS de valider la clé si elle est symétrique et activée.

   L'exemple suivant montre une politique de AWS KMS clé que vous devez appliquer à la clé cible pour le chiffrement. Pour utiliser les exemples de déclarations de politique, *`user input placeholders`* remplacez-les par vos propres informations. Comme toujours, configurez uniquement les autorisations dont vous avez besoin, mais vous devrez AWS KMS fournir des autorisations à au moins un utilisateur pour éviter les erreurs.

```
{
      "Sid": "Allow generate data key access for Fargate tasks.",
      "Effect": "Allow",
      "Principal": { "Service":"fargate.amazonaws.com" },
      "Action": [
        "kms:GenerateDataKeyWithoutPlaintext"
      ],
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:ecs:clusterAccount": [
```

```
            "customerAccountId"
          ],
          "kms:EncryptionContext:aws:ecs:clusterName": [
             "clusterName"
          ]
        }
      },
      "Resource": "*"
    },
    {
      "Sid": "Allow grant creation permission for Fargate tasks.",
      "Effect": "Allow",
      "Principal": { "Service":"fargate.amazonaws.com" },
      "Action": [
        "kms:CreateGrant"
      ],
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:ecs:clusterAccount": [
            "customerAccountId"
          ],
          "kms:EncryptionContext:aws:ecs:clusterName": [
             "clusterName"
          ]
        },
       "ForAllValues:StringEquals": {
          "kms:GrantOperations": [
             "Decrypt"
          ]
       }
      },
      "Resource": "*"
    },
    {
      "Sid": "Allow describe key permission for cluster operator - CreateCluster
 and UpdateCluster.",
      "Effect": "Allow",
      "Principal": { "AWS":"arn:aws:iam::customerAccountId:role/customer-chosen-
role" },
      "Action": [
        "kms:DescribeKey"
      ],
      "Resource": "*"
```

```
    }
```

Les tâches Fargate utilisent les clés de contexte de chiffrement `aws:ecs:clusterName` et pour `aws:ecs:clusterAccount` les opérations cryptographiques effectuées avec la clé. Les clients doivent ajouter ces autorisations pour restreindre l'accès à un and/or groupe de comptes spécifique. Utilisez le nom du cluster et non l'ARN lorsque vous spécifiez le cluster.

Consultez Contexte de chiffrement dans le AWS KMS guide du développeur pour en savoir plus.

Lorsque vous créez ou mettez à jour un cluster, vous avez la possibilité d'utiliser la clé de condition`fargateEphemeralStorageKmsKeyId`. Cette clé de condition permet aux clients de contrôler de manière plus précise les politiques IAM. Les mises à jour de `fargateEphemeralStorageKmsKeyId` configuration ne prennent effet que sur les nouveaux déploiements de services.

L'exemple suivant montre comment autoriser les clients à n'accorder des autorisations qu'à un ensemble spécifique de AWS KMS clés approuvées.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:CreateCluster",
        "ecs:UpdateCluster"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "ecs:fargate-ephemeral-storage-kms-key": "arn:aws:kms:us-
west-2:111122223333:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
        }
      }
    }
  ]
}
```

Voici un exemple de refus des tentatives de suppression de AWS KMS clés déjà associées à un cluster.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": [
        "ecs:CreateCluster",
        "ecs:UpdateCluster"
    ],
    "Resource": "*",
    "Condition": {
      "Null": {
        "ecs:fargate-ephemeral-storage-kms-key": "true"
      }
    }
  }
}
```

Les clients peuvent voir si leurs tâches non gérées ou leurs tâches de service sont chiffrées à l'aide de la clé à l'aide des `describe-services` commandes AWS CLI `describe-tasksdescribe-cluster`, ou.

Pour plus d'informations, consultez la section Clés AWS KMS de condition du guide du AWS KMS développeur.

AWS Management Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Choisissez Clusters dans le menu de navigation de gauche, puis Créez un cluster en haut à droite ou choisissez un cluster existant. Pour un cluster existant, choisissez Mettre à jour le cluster en haut à droite.

3. Dans la section Chiffrement du flux de travail, vous aurez la possibilité de sélectionner votre AWS KMS clé sous Stockage géré et Stockage éphémère Fargate. Vous pouvez également choisir de créer une AWS KMS clé à partir d'ici.
4. Choisissez Créer une fois que vous avez fini de créer votre nouveau cluster ou Mettre à jour, si vous mettiez à jour un cluster existant.

AWS CLI

Voici un exemple de création d'un cluster et de configuration de votre stockage éphémère Fargate à l'aide AWS CLI du (remplacez *red* les valeurs par les vôtres) :

```
aws ecs create-cluster --cluster clusterName \
--configuration '{"managedStorageConfiguration":
{"fargateEphemeralStorageKmsKeyId":"arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"}}'
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:012345678901:cluster/clusterName",
        "clusterName": "clusterName",
        "configuration": {
            "managedStorageConfiguration": {
                "fargateEphemeralStorageKmsKeyId": "arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
            }
        },
        "status": "ACTIVE",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [],
        "capacityProviders": [],
        "defaultCapacityProviderStrategy": []
    },
    "clusterCount": 5
}
```

AWS CloudFormation

Voici un exemple de modèle de création d'un cluster et de configuration de votre stockage éphémère Fargate à l'aide AWS CloudFormation du (remplacez *red* les valeurs par les vôtres) :

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  MyCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: "clusterName"
      Configuration:
        ManagedStorageConfiguration:
          FargateEphemeralStorageKmsKeyId: "arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Gestion des AWS KMS clés pour le stockage éphémère Fargate pour Amazon ECS

Après avoir créé ou importé votre AWS KMS clé pour chiffrer votre stockage éphémère Fargate, vous le gérez de la même manière que n'importe quelle autre clé. AWS KMS

Rotation automatique des AWS KMS touches

Vous pouvez activer la rotation automatique des touches ou les faire pivoter manuellement. La rotation automatique des clés fait pivoter la clé pour vous chaque année en générant du nouveau matériel cryptographique pour la clé. AWS KMS enregistre également toutes les versions précédentes du matériel cryptographique, de sorte que vous serez en mesure de déchiffrer toutes les données qui utilisaient les versions de clé antérieures. Aucun matériau pivoté ne sera supprimé AWS KMS tant que vous n'aurez pas supprimé la clé.

La rotation automatique des touches est facultative et peut être activée ou désactivée à tout moment.

Désactivation ou révocation des clés AWS KMS

Si vous désactivez une clé d'entrée gérée par le client AWS KMS, cela n'a aucun impact sur l'exécution des tâches et celles-ci continuent de fonctionner tout au long de leur cycle de vie. Si une nouvelle tâche utilise la clé désactivée ou révoquée, elle échoue car elle ne peut pas accéder à la clé. Vous devez définir une CloudWatch alarme ou une alarme similaire pour vous assurer qu'une clé désactivée ne soit jamais nécessaire pour déchiffrer des données déjà chiffrées.

Supprimer des AWS KMS clés

La suppression de clés doit toujours être une solution de dernier recours et ne doit être effectuée que si vous êtes certain que la clé supprimée ne sera plus jamais nécessaire. Les nouvelles tâches qui tentent d'utiliser la clé supprimée échoueront car elles ne pourront pas y accéder. AWS KMS conseille de désactiver une clé au lieu de la supprimer. Si vous pensez qu'il est nécessaire de supprimer une clé, nous vous suggérons de la désactiver d'abord et de configurer une CloudWatch alarme pour vous assurer qu'elle n'est pas nécessaire. Si vous supprimez une clé, vous AWS KMS disposez d'au moins sept jours pour changer d'avis.

Audit de l'accès aux AWS KMS clés

Vous pouvez utiliser CloudTrail les journaux pour vérifier l'accès à votre AWS KMS clé. Vous pouvez vérifier les AWS KMS opérations `CreateGrantGenerateDataKeyWithoutPlaintext`, et`Decrypt`. Ces opérations affichent également le `aws:ecs:clusterAccount` et dans le `aws:ecs:clusterName` cadre de `EncryptionContext` la connexion CloudTrail.

Voici des exemples d' CloudTrail événements pour`GenerateDataKeyWithoutPlaintext`,`GenerateDataKeyWithoutPlaintext (DryRun)`, `CreateGrantCreateGrant (DryRun)`, et `RetireGrant` (remplacez les *`red`* valeurs par les vôtres).

GenerateDataKeyWithoutPlaintext

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ec2-frontend-api.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:13Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ec2-frontend-api.amazonaws.com",
    "userAgent": "ec2-frontend-api.amazonaws.com",
    "requestParameters": {
        "numberOfBytes": 64,
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "encryptionContext": {
            "aws:ecs:clusterAccount": "account-id",
            "aws:ebs:id": "vol-xxxxxxx",
```

```
            "aws:ecs:clusterName": "cluster-name"
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

GenerateDataKeyWithoutPlaintext (DryRun)

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "fargate.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "fargate.amazonaws.com",
    "userAgent": "fargate.amazonaws.com",
    "errorCode": "DryRunOperationException",
    "errorMessage": "The request would have succeeded, but the DryRun option is
 set.",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "dryRun": true,
```

```
        "numberOfBytes": 64,
        "encryptionContext": {
            "aws:ecs:clusterAccount": "account-id",
            "aws:ecs:clusterName": "cluster-name"
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

CreateGrant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ec2-frontend-api.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:13Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ec2-frontend-api.amazonaws.com",
    "userAgent": "ec2-frontend-api.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "granteePrincipal": "fargate.us-west-2.amazonaws.com",
```

```
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:ecs:clusterAccount": "account-id",
                "aws:ebs:id": "vol-xxxx",
                "aws:ecs:clusterName": "cluster-name"
            }
        },
        "retiringPrincipal": "ec2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855",
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
    },
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

CreateGrant (DryRun)

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "fargate.amazonaws.com"
```

```
    },
    "eventTime": "2024-04-23T18:08:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "fargate.amazonaws.com",
    "userAgent": "fargate.amazonaws.com",
    "errorCode": "DryRunOperationException",
    "errorMessage": "The request would have succeeded, but the DryRun option is
 set.",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "granteePrincipal": "fargate.us-west-2.amazonaws.com",
        "dryRun": true,
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:ecs:clusterAccount": "account-id",
                "aws:ecs:clusterName": "cluster-name"
            }
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

RetireGrant


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2024-04-20T18:37:38Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "RetireGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
    },
    "additionalEventData": {
        "grantId":
 "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855"
    },
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS

AWS est responsable de la maintenance de l'infrastructure sous-jacente de AWS Fargate. AWS détermine à quel moment une révision de version de plate-forme doit être remplacée par une nouvelle révision de l'infrastructure. C'est ce que l'on appelle la retraite des tâches. AWS envoie une notification de retrait de tâche lorsqu'une révision de version de plateforme est supprimée. Nous mettons régulièrement à jour les versions de nos plateformes prises en charge pour introduire une nouvelle révision contenant des mises à jour du logiciel d'exécution Fargate et des dépendances sous-jacentes telles que le système d'exploitation et le runtime du conteneur. Une fois qu'une nouvelle révision est disponible, nous retirons l'ancienne version afin de garantir que toutes les charges de travail des clients soient exécutées sur la version la plus récente de la version de la plateforme Fargate. Lorsqu'une révision est retirée, toutes les tâches exécutées sur cette révision sont arrêtées.

Les tâches Amazon ECS peuvent être classées en tant que tâches de service ou tâches autonomes. Les tâches de service sont déployées dans le cadre d'un service et contrôlées par le calendrier Amazon ECS. Pour de plus amples informations, veuillez consulter Services Amazon ECS. Les tâches autonomes sont des tâches démarrées par l'`RunTask`API Amazon ECS, soit directement, soit par un planificateur externe, telles que les tâches planifiées (lancées par Amazon EventBridge), AWS Batch ou. AWS Step Functions Vous n'avez aucune action à effectuer en réponse au retrait des tâches pour vos tâches de service, car le planificateur Amazon ECS remplace automatiquement les tâches.

Pour les tâches autonomes, il se peut que vous deviez effectuer une manipulation supplémentaire en réponse à la suppression des tâches. Pour de plus amples informations, veuillez consulter Amazon ECS peut-il gérer automatiquement des tâches autonomes ?.

Pour les tâches de service, il n'est pas nécessaire de prendre des mesures pour les supprimer, sauf si vous souhaitez remplacer ces tâches au AWS préalable. Lorsque le planificateur Amazon ECS arrête les tâches, il les utilise `maximumPercent` et lance une nouvelle tâche afin de maintenir le nombre souhaité pour le service. Suivez les meilleures pratiques pour minimiser l'impact de la suppression des tâches. La `maximumPercent` valeur par défaut pour un service utilisant le planificateur de services REPLICA est de 200 %. Par conséquent, lorsqu'il AWS Fargate commence à retirer des tâches, Amazon ECS planifie d'abord une nouvelle tâche, puis attend qu'elle soit exécutée avant de retirer une ancienne tâche. Lorsque vous définissez la `maximumPercent` valeur sur 100 %, Amazon ECS arrête d'abord la tâche, puis la remplace.

Pour le retrait d'une tâche autonome, AWS arrête la tâche à la date de retrait de la tâche ou après cette date. Amazon ECS ne lance pas de tâche de remplacement lorsqu'une tâche est arrêtée. Si vous souhaitez que ces tâches continuent de s'exécuter, vous devez arrêter les tâches en cours d'exécution et lancer une tâche de remplacement avant l'heure indiquée dans la notification. Nous recommandons donc aux clients de surveiller l'état des tâches autonomes et, si nécessaire, d'implémenter une logique pour remplacer les tâches interrompues.

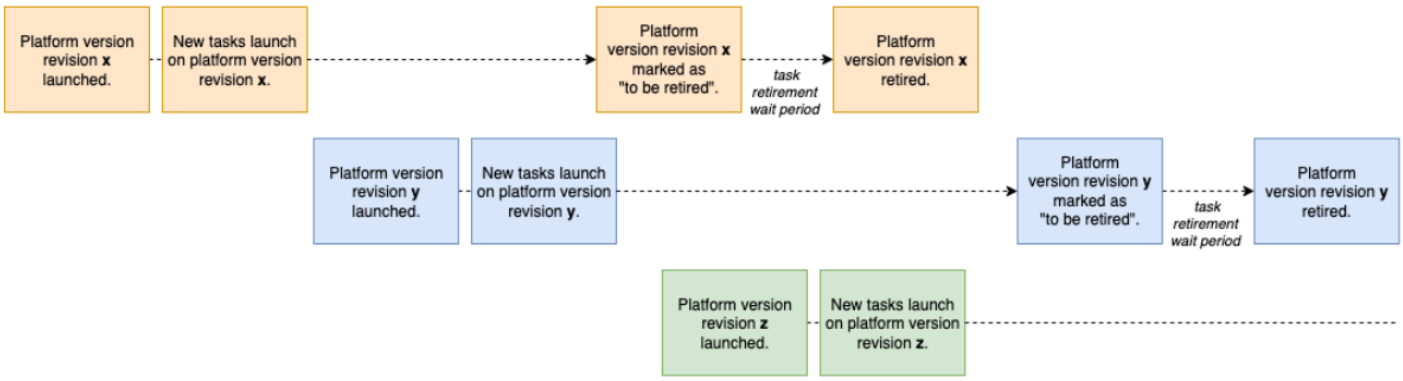
Lorsqu'une tâche est arrêtée dans l'un des scénarios ci-dessus, vous pouvez exécuter `describe-tasks`. La `stoppedReason` de la réponse est `ECS is performing maintenance on the underlying infrastructure hosting the task`.

La maintenance des tâches s'applique lorsqu'une nouvelle version de la plateforme doit être remplacée par une nouvelle révision. En cas de problème avec un hôte Fargate sous-jacent, Amazon ECS remplace l'hôte sans préavis de retrait de tâche.

Aperçu de l'avis de retrait des tâches

Lorsqu' AWS une révision de version de plate-forme doit être supprimée, nous identifions toutes les tâches en cours d'exécution sur cette révision de version de plate-forme dans toutes les régions. Nous envoyons ensuite une notification par compte et par région, mettant en évidence les tâches ou services concernés et indiquant la date à laquelle les départs à la retraite commenceront à avoir lieu.

L'illustration suivante montre le cycle de vie d'une révision de version de la plateforme Fargate, depuis le lancement d'une nouvelle révision jusqu'au retrait de la révision de la plate-forme.



Les informations suivantes fournissent des informations détaillées.

- Après le lancement d'une nouvelle révision de la version de la plateforme, toutes les nouvelles tâches sont planifiées sur cette révision.

- Les tâches existantes qui ont été planifiées et exécutées restent dans la version dans laquelle elles ont été initialement placées pendant toute la durée de la tâche et ne sont pas migrées vers la nouvelle révision.
- Les nouvelles tâches, par exemple dans le cadre d'une mise à jour d'un service ou de la suppression de tâches Fargate, sont placées sur la dernière version de la plateforme disponible au moment du lancement.

Les notifications de retrait de tâches sont envoyées via le tableau de AWS Health bord ainsi que par e-mail à l'adresse e-mail enregistrée et incluent les informations suivantes :

- Date de retrait de la tâche : la tâche est arrêtée à cette date ou après.
- Pour les tâches autonomes, IDs les tâches.
- Pour les tâches de service, l'ID du cluster sur lequel le service s'exécute et IDs celui du service.
- Les prochaines étapes que vous devez suivre.

En règle générale, nous envoyons une notification pour les tâches de service et les tâches autonomes dans chaque Région AWS cas. Toutefois, dans certains cas, vous pouvez recevoir plusieurs événements pour chaque type de tâche, par exemple lorsqu'un trop grand nombre de tâches dépassent les limites fixées par nos mécanismes de notification pour être supprimées.

Vous pouvez identifier les tâches planifiées pour le retrait en procédant ainsi :

- Le AWS Health Dashboard

  AWS Health les notifications peuvent être envoyées via Amazon EventBridge vers un système de stockage d'archives tel qu'Amazon Simple Storage Service, effectuer des actions automatisées telles que l'exécution d'une AWS Lambda fonction, ou vers d'autres systèmes de notification tels qu'Amazon Simple Notification Service. Pour plus d'informations, consultez la section Surveillance AWS Health des événements avec Amazon EventBridge. Pour un exemple de configuration permettant d'envoyer des notifications à Amazon Chime, Slack ou Microsoft Teams, consultez le référentiel AWS Health Aware sur. GitHub

  Voici un exemple d' EventBridge événement.

```
{
    "version": "0",
    "id": "3c268027-f43c-0171-7425-1d799EXAMPLE",
```

```
    "detail-type": "AWS Health Event",
    "source": "aws.health",
    "account": "123456789012",
    "time": "2023-08-16T23:18:51Z",
    "region": "us-east-1",
    "resources": [
        "cluster|service",
        "cluster|service"
    ],
    "detail": {
        "eventArn": "arn:aws:health:us-east-1::event/ECS/
AWS_ECS_TASK_PATCHING_RETIREMENT/AWS_ECS_TASK_PATCHING_RETIREMENT_test1",
        "service": "ECS",
        "eventScopeCode": "ACCOUNT_SPECIFIC",
        "communicationId":
 "7988399e2e6fb0b905ddc88e0e2de1fd17e4c9fa60349577446d95a18EXAMPLE",
        "lastUpdatedTime": "Wed, 16 Aug 2023 23:18:52 GMT",
        "eventRegion": "us-east-1",
        "eventTypeCode": "AWS_ECS_TASK_PATCHING_RETIREMENT",
        "eventTypeCategory": "scheduledChange",
        "startTime": "Wed, 16 Aug 2023 23:18:51 GMT",
        "endTime": "Fri, 18 Aug 2023 23:18:51 GMT",
        "eventDescription": [
            {
                "language": "en_US",
                "latestDescription": "\\nA software update has been deployed to
 Fargate which includes CVE patches or other critical patches. No action is required
 on your part. All new tasks launched automatically uses the latest software
 version. For existing tasks, your tasks need to be restarted in order for these
 updates to apply. Your tasks running as part of the following ECS Services will
 be automatically updated beginning Wed, 16 Aug 2023 23:18:51 GMT.\\n\\nAfter Wed,
 16 Aug 2023 23:18:51 GMT, the ECS scheduler will gradually replace these tasks,
 respecting the deployment settings for your service. Typically, services should
 see little to no interruption during the update and no action is required. When AWS
 stops tasks, AWS uses the minimum healthy percent (1) and launches a new task in
 an attempt to maintain the desired count for the service. By default, the minimum
 healthy percent of a service is 100 percent, so a new task is started first before
 a task is stopped. Service tasks are routinely replaced in the same way when
 you scale the service or deploy configuration changes or deploy task definition
 revisions. If you would like to control the timing of this restart you can update
 the service before Wed, 16 Aug 2023 23:18:51 GMT, by running the update-service
 command from the ECS command-line interface specifying force-new-deployment for
 services using Rolling update deployment type. For example:\\n\\n$ aws ecs update-
service -service service_name \\\n--cluster cluster_name -force-new-deployment\
```

```
\n\\nFor services using Blue/Green deployment type with AWS CodeDeploy:\\nPlease
 refer to create-deployment document (2) and create new deployment using same task
 definition revision.\\n\\nFor further details on ECS deployment types, please
 refer to ECS Deployment Developer Guide (1).\\nFor further details on Fargate's
 update process, please refer to the AWS Fargate User Guide (3).\\nIf you have
 any questions or concerns, please contact AWS Support (4).\\n\\n(1) https://
docs.aws.amazon.com/AmazonECS/latest/developerguide/deployment-types.html\\n(2)
 https://docs.aws.amazon.com/cli/latest/reference/deploy/create-deployment.html\\n(3)
 https://docs.aws.amazon.com/AmazonECS/latest/userguide/task-maintenance.html\\n(4)
 https://aws.amazon.com/support\\n\\nA list of your affected resources(s) can be
 found in the 'Affected resources' tab in the 'Cluster/ Service' format in the AWS
 Health Dashboard. \\n\\n"
            }
        ],
      "affectedEntities": [
                {
                    "entityValue": "arn:aws:ecs:eu-west-1:111222333444:task/
examplecluster/00805ce1d81940b5a37398e5a2c23333"
                },
                {
                    "entityValue": "arn:aws:ecs:eu-west-1:111222333444:task/
examplecluster/00805ce1d81940b5a37398e5a2c25555"
                }
            }
        ]
    }
}
```

- E-mails

  Un e-mail est envoyé à l'adresse e-mail enregistrée pour l' Compte AWS identifiant.

Pour plus d'informations sur la façon de se préparer à la mise hors service des tâches, consultezPréparez-vous à la AWS suppression des tâches Fargate sur Amazon ECS .

Puis-je me désinscrire de la mise hors service des tâches ?

Non Dans le cadre du modèle de responsabilité AWS partagée, AWS est responsable de la gestion et de la maintenance de l'infrastructure sous-jacente pour AWS Fargate. Cela inclut l'exécution de mises à jour périodiques de la plateforme pour garantir la sécurité et la stabilité. Ces mises à jour sont automatiquement appliquées par les clients AWS et ne peuvent pas s'y soustraire. Il s'agit d'un avantage clé de l'utilisation d'une AWS Fargate par rapport à l'exécution de vos charges de travail

sur EC2 des instances, la responsabilité de la maintenance de la plate-forme sous-jacente étant prise en charge par AWS. Ce modèle vous permet de vous concentrer sur vos applications plutôt que sur la maintenance de l'infrastructure. En appliquant automatiquement ces mises à jour de la plateforme, AWS vous êtes en mesure de maintenir l' up-to-dateenvironnement Fargate et de le sécuriser, sans qu'aucune action ne soit requise de votre part en tant que client. Cela permet de fournir un environnement conteneurisé fiable et sécurisé pour exécuter vos charges de travail sur Fargate.

Puis-je recevoir des notifications de retrait de tâches par le biais d'autres AWS services ?

AWS envoie une notification de retrait de tâche au AWS Health Dashboard et au contact e-mail principal sur le Compte AWS. AWS Health Dashboard Il fournit un certain nombre d'intégrations dans d'autres AWS services, notamment EventBridge. Vous pouvez l'utiliser EventBridge pour automatiser la visibilité des notifications (par exemple, transférer le message vers un ChatOps outil). Pour plus d'informations, voir Présentation de la solution : capture des notifications de retrait de tâches.

Puis-je modifier le retrait d'une tâche une fois qu'il a été planifié ?

Non Le calendrier est basé sur le temps d'attente pour la fin des tâches, qui est par défaut de 7 jours. Si vous avez besoin de plus de temps, vous pouvez choisir de configurer le délai d'attente à 14 jours. Pour de plus amples informations, veuillez consulter Étape 2 : Capturez les notifications de retrait des tâches pour alerter les équipes et prendre des mesures. La modification apportée à cette configuration s'applique aux mises à la retraite qui seront planifiées dans le futur. Les départs à la retraite prévus actuellement ne sont pas concernés. Si vous avez d'autres préoccupations, contactez Support.

Comment Amazon ECS gère-t-il les tâches faisant partie d'un service ?

Pour les tâches de service, il n'est pas nécessaire de prendre des mesures en réponse à la suppression des tâches, sauf si vous souhaitez remplacer ces tâches au AWS préalable. Lorsque le planificateur Amazon ECS arrête les tâches, il utilise le pourcentage de santé minimum et lance une nouvelle tâche afin de maintenir le nombre souhaité pour le service. Pour minimiser l'impact du retrait des tâches Fargate, les charges de travail doivent être déployées conformément aux meilleures pratiques d'Amazon ECS. Par exemple, lors du déploiement d'une application sans état en tant que service Amazon ECS, tel qu'un serveur Web ou API, les clients doivent déployer plusieurs répliques de tâches et définir la valeur minimumHealthyPercent à 100 %. Par défaut, le pourcentage de santé minimum d'un service est de 100 %. Par conséquent, lorsque Fargate commence à retirer

des tâches, Amazon ECS planifie d'abord une nouvelle tâche et attend qu'elle soit exécutée, avant de retirer une ancienne tâche. Les tâches de service sont régulièrement remplacées dans le cadre du retrait des tâches, de la même manière que lorsque vous adaptez le service, déployez des modifications de configuration ou déployez des révisions de définitions de tâches. Pour préparer le processus de retrait des tâches, voir Préparez-vous à la AWS suppression des tâches Fargate sur Amazon ECS .

Amazon ECS peut-il gérer automatiquement des tâches autonomes ?

Non AWS ne peut pas créer de tâche de remplacement pour les tâches autonomes démarrées par `RunTask` des tâches planifiées (par exemple via le EventBridge planificateur) ou. AWS Batch AWS Step Functions Amazon ECS gère uniquement les tâches faisant partie d'un service.

Préparez-vous à la AWS suppression des tâches Fargate sur Amazon ECS

Afin de préparer le retrait des tâches, effectuez les opérations suivantes :

1. Définissez la période d'attente pour la mise hors service des tâches.
2. Capturez les notifications de retrait des tâches pour avertir les membres de l'équipe.
3. Vous ne pouvez pas contrôler le moment exact du retrait d'une tâche, mais vous pouvez contrôler le remplacement des tâches en mettant à jour le service avec l'option de déploiement forcé.

Étape 1 : définir le temps d'attente des tâches

Vous pouvez configurer l'heure à laquelle Fargate commence le retrait des tâches. Pour les charges de travail qui nécessitent l'application immédiate des mises à jour, choisissez le paramètre immédiat (`0`). Lorsque vous avez besoin de davantage de contrôle, par exemple lorsqu'une tâche ne peut être arrêtée que pendant une certaine période, configurez l'option 7 jours (`7`) ou 14 jours (`14`).

Nous vous recommandons de choisir une période d'attente plus courte afin de pouvoir accéder plus rapidement aux nouvelles versions de plateforme.

Configurez la période d'attente en exécutant `put-account-setting-default` ou `put-account-setting` en tant qu'utilisateur root ou administrateur. Utilisez l'option `fargateTaskRetirementWaitPeriod` pour le `name` et l'option `value` définie sur l'une des valeurs suivantes :

- `0`- AWS envoie la notification et commence immédiatement à supprimer les tâches concernées.

- 7- AWS envoie la notification et attend 7 jours calendaires avant de commencer à supprimer les tâches concernées.
- 14 : AWS envoie la notification et attend 14 jours calendaires avant de commencer à retirer les tâches concernées.

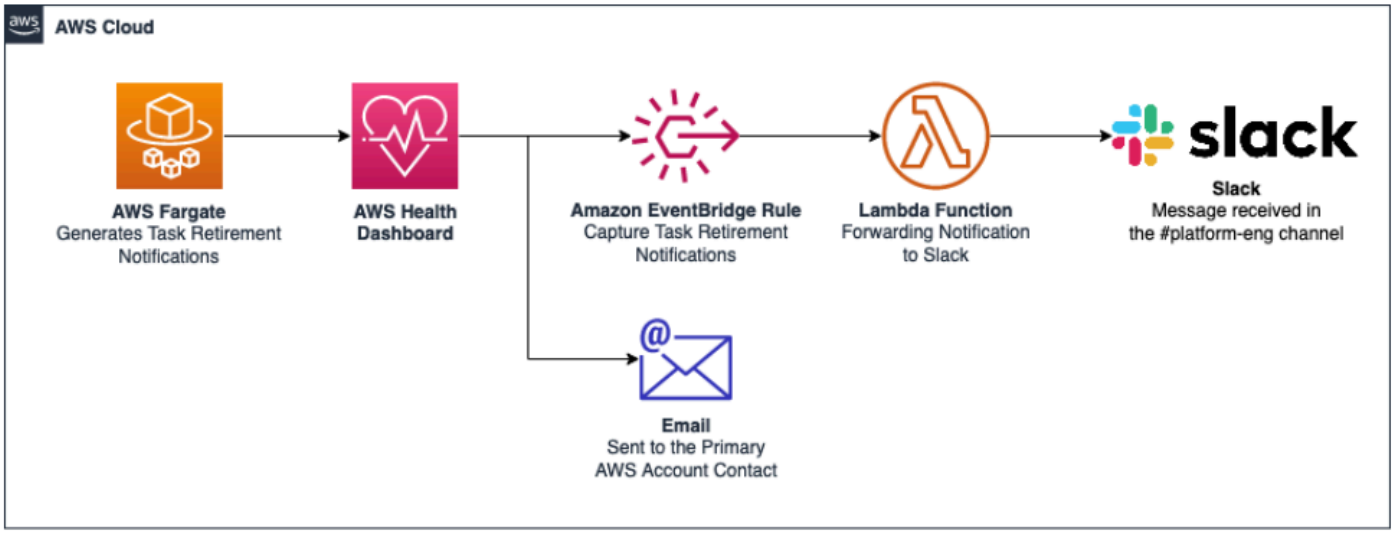
La durée par défaut est de 7 jours.

Pour plus d'informations, consultez put-account-setting-defaultet consultez le put-account-settingmanuel Amazon Elastic Container Service API Reference.

Étape 2 : Capturez les notifications de retrait des tâches pour alerter les équipes et prendre des mesures

En cas de retrait imminent d'une tâche, AWS envoie une notification de retrait de tâche au AWS Health tableau de bord et au contact e-mail principal sur le Compte AWS. Le AWS Health tableau de bord propose un certain nombre d'intégrations dans d'autres AWS services, notamment Amazon EventBridge. Vous pouvez l'utiliser EventBridge pour créer des automatisations à partir d'une notification de retrait de tâche, par exemple en augmentant la visibilité du prochain retrait en transférant le message vers un ChatOps outil. AWS Health Aware est une ressource qui montre la puissance du AWS Health tableau de bord et montre comment les notifications peuvent être distribuées au sein d'une organisation. Vous pouvez transférer une notification de retrait de tâche vers une application de chat, telle que Slack.

L'illustration suivante montre la vue d'ensemble de la solution.



Les informations suivantes fournissent des informations détaillées.

- Fargate envoie la notification de retrait de la tâche au tableau de bord. AWS Health
- Le AWS Health tableau de bord envoie un e-mail au contact e-mail principal sur le Compte AWS, et le notifie EventBridge.
- EventBridge possède une règle qui capture la notification de départ à la retraite.

  La règle qui recherche les événements avec le type de détail de l'événement : `"AWS Health Event" and the Event Detail Type Code: "AWS_ECS_TASK_PATCHING_RETIREMENT"`
- La règle déclenche une fonction Lambda qui transmet les informations à Slack à l'aide d'un webhook entrant Slack. Pour plus d'informations, consultez la section Webhooks entrants.

Pour un exemple de code, consultez Capture des notifications de retrait de AWS Fargate tâches sur Github.

Étape 3 : Contrôler le remplacement des tâches

Vous ne pouvez pas contrôler le moment exact du retrait d'une tâche, mais vous pouvez définir un temps d'attente. Si vous souhaitez contrôler le remplacement des tâches selon votre propre calendrier, vous pouvez enregistrer l'avis de retrait des tâches afin de comprendre d'abord la date de fin des tâches. Vous pouvez ensuite redéployer votre service pour lancer des tâches de remplacement, ainsi que pour remplacer toutes les tâches autonomes. Pour les services utilisant un déploiement progressif, vous devez mettre à jour le service à l'aide `update-service` de l'`force-deployment`option avant l'heure de début de la mise hors service.

L'`update-service`exemple suivant utilise l'`force-deployment`option.

```
aws ecs update-service —-service service_name \
    --cluster cluster_name \
     --force-new-deployment
```

Pour les services qui utilisent le blue/green déploiement, vous devez créer un nouveau déploiement dans AWS CodeDeploy. Pour plus d'informations sur la façon de créer le déploiement, voir create-deployment dans la AWS Command Line Interface référence.

Régions prises en charge pour Amazon ECS sur AWS Fargate

Vous pouvez utiliser les tableaux suivants pour vérifier le support régional pour les conteneurs Linux sur AWS Fargate et les conteneurs Windows sur Fargate. AWS

Conteneurs Linux sur AWS Fargate

Les conteneurs Linux Amazon ECS activés AWS Fargate sont pris en charge dans les versions suivantes Régions AWS. La zone de disponibilité prise en charge IDs est indiquée le cas échéant.

Nom de la région	Région
USA Est (Ohio)	us-east-2
USA Est (Virginie du Nord)	us-east-1
USA Ouest (Californie du Nord)	us-west-1 (`usw1-az1` et `usw1-az3` uniquement)
USA Ouest (Oregon)	us-west-2
Canada (Centre)	ca-central-1
Canada Ouest (Calgary)	ca-west-1
Mexique (centre)	mx-central-1
Afrique (Le Cap)	af-south-1
Asie-Pacifique (Hong Kong)	ap-east-1
Asie-Pacifique (Mumbai)	ap-south-1
Asie-Pacifique (Tokyo)	ap-northeast-1 (`apne1-az1` , `apne1-az2` et `apne1-az4` uniquement)
Asie-Pacifique (Séoul)	ap-northeast-2
Asie-Pacifique (Osaka)	ap-northeast-3
Asie-Pacifique (Hyderabad)	ap-south-2

Nom de la région	Région
Asie-Pacifique (Singapour)	ap-southeast-1
Asie-Pacifique (Sydney)	ap-southeast-2
Asie-Pacifique (Thaïlande)	ap-sud-est 7
Asie-Pacifique (Jakarta)	ap-southeast-3
Asie-Pacifique (Melbourne)	ap-southeast-4
Asie-Pacifique (Malaisie)	ap-southeast-5
Canada (Centre)	ca-central-1
Canada Ouest (Calgary)	ca-west-1
Chine (Beijing)	cn-north-1 (`cnn1-az1` et `cnn1-az2` uniquement)
Chine (Ningxia)	cn-northwest-1
Europe (Francfort)	eu-central-1
Europe (Zurich)	eu-central-2
Europe (Irlande)	eu-west-1
Europe (Londres)	eu-west-2
Europe (Paris)	eu-west-3
Europe (Milan)	eu-south-1
Europe (Espagne)	eu-south-2
Europe (Stockholm)	eu-north-1
Amérique du Sud (São Paulo)	sa-east-1
Israël (Tel Aviv)	il-central-1

Nom de la région	Région
Moyen-Orient (Bahreïn)	me-south-1
Moyen-Orient (EAU)	me-central-1
AWS GovCloud (USA Est)	us-gov-east-1
AWS GovCloud (US-Ouest)	us-gov-west-1

Conteneurs Windows sur AWS Fargate

Les conteneurs Windows Amazon ECS activés AWS Fargate sont pris en charge dans les versions suivantes Régions AWS. La zone de disponibilité prise en charge IDs est indiquée le cas échéant.

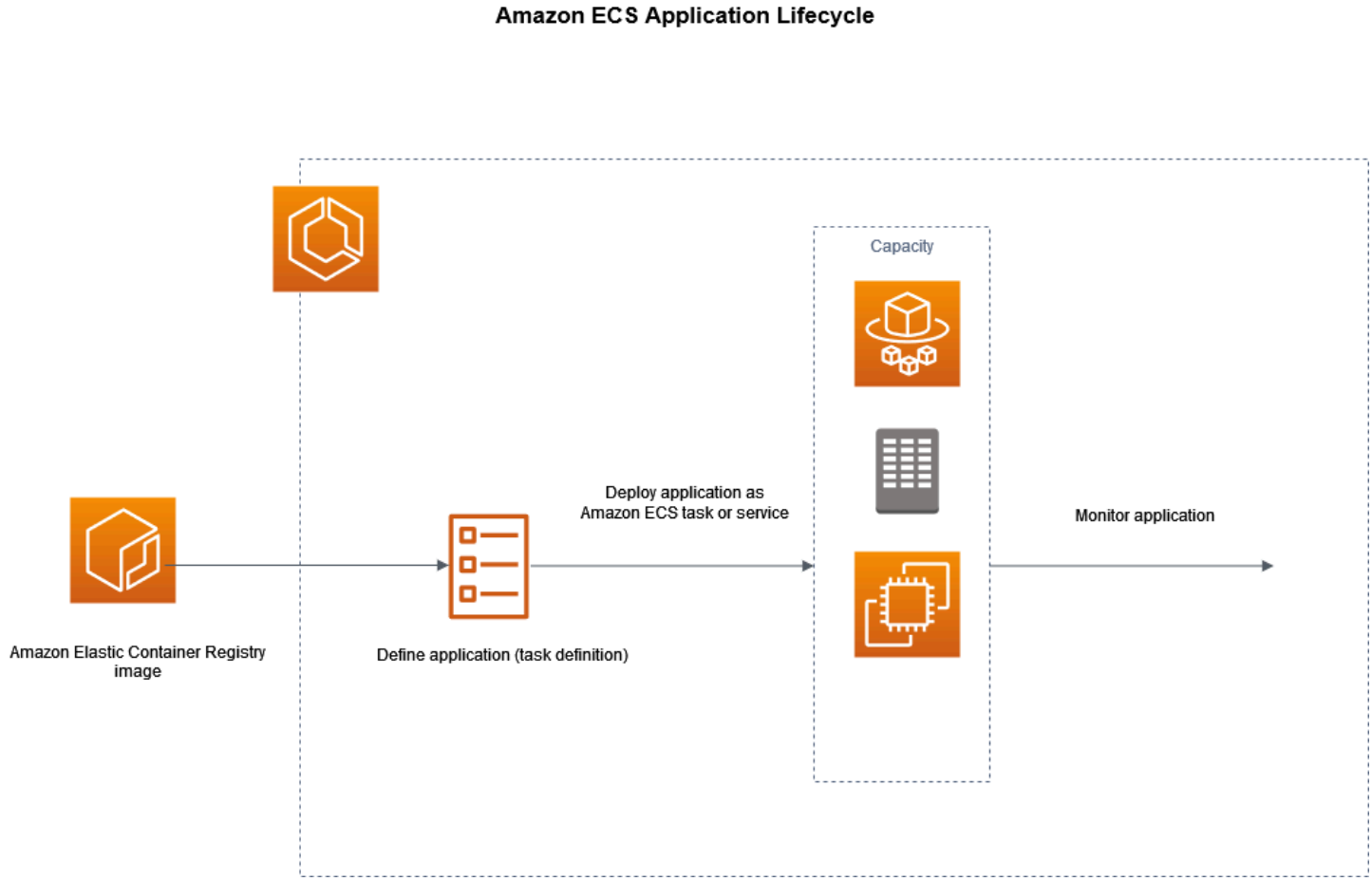
Nom de la région	Région
USA Est (Ohio)	us-east-2
USA Est (Virginie du Nord)	us-east-1 (`use1-az1`, `use1-az2`, `use1-az4`, `use1-az5` et `use1-az6` uniquement)
USA Ouest (Californie du Nord)	us-west-1 (`usw1-az1` et `usw1-az3` uniquemen t)
USA Ouest (Oregon)	us-west-2
Afrique (Le Cap)	af-south-1
Asie-Pacifique (Hong Kong)	ap-east-1
Asie-Pacifique (Mumbai)	ap-south-1
Asie-Pacifique (Hyderabad)	ap-south-2
Asie-Pacifique (Osaka)	ap-northeast-3
Asie-Pacifique (Séoul)	ap-northeast-2
Asie-Pacifique (Singapour)	ap-southeast-1

Nom de la région	Région
Asie-Pacifique (Sydney)	ap-southeast-2
Asie-Pacifique (Melbourne)	ap-southeast-4
Asie-Pacifique (Malaisie)	ap-southeast-5
Asie-Pacifique (Tokyo)	ap-northeast-1 (`apne1-az1` , `apne1-az2` et `apne1-az4` uniquement)
Canada (Centre)	ca-central-1 (`cac1-az1` et `cac1-az2` uniquement)
Canada Ouest (Calgary)	ca-west-1
Chine (Beijing)	cn-north-1 (`cnn1-az1` et `cnn1-az2` uniquement)
Chine (Ningxia)	cn-northwest-1
Europe (Francfort)	eu-central-1
Europe (Zurich)	eu-central-2
Europe (Irlande)	eu-west-1
Europe (Londres)	eu-west-2
Europe (Paris)	eu-west-3
Europe (Milan)	eu-south-1
Europe (Espagne)	eu-south-2
Europe (Stockholm)	eu-north-1
Amérique du Sud (São Paulo)	sa-east-1
Israël (Tel Aviv)	il-central-1
Moyen-Orient (EAU)	me-central-1

Nom de la région	Région
Moyen-Orient (Bahreïn)	me-south-1

Concevez votre solution pour Amazon ECS

Le schéma suivant montre le cycle de vie de l'application et son fonctionnement avec les composants Amazon ECS.



Vous devez concevoir vos applications de manière à ce qu'elles puissent s'exécuter sur des conteneurs. Un conteneur est une unité standardisée de développement logiciel qui contient tout ce dont votre application logicielle a besoin pour être exécutée. Cela inclut le code, l'exécution, les outils système et les bibliothèques système pertinents. Les conteneurs sont créés à partir d'un modèle en lecture seule appelé image. Les images sont généralement créées à partir d'un fichier Dockerfile. Un Dockerfile est un fichier en texte brut qui contient les instructions pour créer un conteneur. Après leur création, ces images sont stockées dans un registre, comme Amazon ECR, d'où elles peuvent être téléchargées.

Après avoir créé et stocké votre image, vous créez une définition de tâche Amazon ECS. Une définition de tâche est le plan de votre application. Il s'agit d'un fichier texte au format JSON qui décrit les paramètres et un ou plusieurs conteneurs qui forment votre application. Par exemple, vous

pouvez l'utiliser pour spécifier l'image et les paramètres du système d'exploitation, les conteneurs à utiliser, les ports à ouvrir pour votre application et les volumes de données à utiliser avec les conteneurs dans la tâche. Les paramètres spécifiques disponibles pour votre définition de tâche dépendent des besoins de votre application spécifique.

Après avoir défini votre définition de tâche, vous la déployez sous forme de service ou de tâche sur votre cluster. Un cluster est un regroupement logique de tâches ou de services qui s'exécute sur l'infrastructure de capacité enregistrée dans un cluster.

Une tâche est l'instanciation d'une définition de tâche au sein d'un cluster. Vous pouvez exécuter une tâche autonome ou exécuter une tâche dans le cadre d'un service. Vous pouvez utiliser un service Amazon ECS service pour exécuter et gérer simultanément le nombre souhaité de tâches dans un cluster Amazon ECS. Le principe est le suivant : si l'une de vos tâches échoue ou s'arrête pour une raison quelconque, le planificateur de service d'Amazon ECS service lance une autre instance en fonction de votre définition de tâche. Il procède ainsi pour le remplacer et donc maintenir le nombre de tâches souhaité dans le service.

L'agent de conteneur s'exécute sur chaque instance de conteneur dans un cluster Amazon ECS. L'agent envoie à Amazon ECS des informations relatives aux tâches en cours d'exécution et à l'utilisation des ressources de vos conteneurs. Il démarre et arrête les tâches lorsqu'il reçoit une requête de la part d'Amazon ECS.

Une fois la tâche ou le service déployés, vous pouvez utiliser l'un des outils suivants pour surveiller votre déploiement et votre application :

- CloudWatch
- Surveillance d'exécution

Capacité

La capacité est l'infrastructure dans laquelle vos conteneurs fonctionnent. Les options sont les suivantes :

- EC2 Instances Amazon
- Sans serveur ()AWS Fargate
- Machines virtuelles (VM) ou serveurs sur site

Vous spécifiez l'infrastructure lorsque vous créez un cluster. Vous spécifiez également le type d'infrastructure lorsque vous enregistrez une définition de tâche. La définition de la tâche désigne l'infrastructure sous le nom de « type de lancement ». Vous utilisez également le type de lancement lorsque vous exécutez une tâche autonome ou que vous déployez un service. Pour plus d'informations sur les options de type de lancement, consultezTypes de lancement Amazon ECS.

Points de terminaison de service

Le point de terminaison du service est l'URL du point d'entrée d'Amazon ECS que vous utilisez pour vous connecter au service par programmation à l'aide du protocole Internet version 4 (IPv4) ou du protocole Internet version 6 (IPv6). Par défaut, les demandes que vous effectuez pour vous connecter à Amazon ECS par programmation utilisent des points de terminaison de service qui ne prennent en charge que les demandes. IPv4 Pour vous connecter par programmation à Amazon ECS à l'aide de l'une IPv4 ou de plusieurs IPv6 requêtes, vous pouvez utiliser un point de terminaison à double pile. Pour plus d'informations sur l'utilisation d'un point de terminaison à double pile, consultezUtilisation des points de terminaison à double pile Amazon ECS.

Réseaux

AWS les ressources sont créées dans des sous-réseaux. Lorsque vous utilisez EC2 des instances, Amazon ECS lance les instances dans le sous-réseau que vous spécifiez lorsque vous créez un cluster. Vos tâches s'exécutent dans le sous-réseau de l'instance. Pour Fargate ou les machines virtuelles locales, vous devez spécifier le sous-réseau lorsque vous exécutez une tâche ou créez un service.

Selon votre application, le sous-réseau peut être un sous-réseau privé ou public et le sous-réseau peut se trouver dans l'une des ressources suivantes : AWS

- Zones de disponibilité
- Zones locales
- Zones Wavelength
- Régions AWS
- AWS Outposts

Pour plus d'informations, consultez Applications Amazon ECS dans des sous-réseaux partagés, des zones locales et des zones de longueur d'onde ou Amazon Elastic Container Service sur AWS Outposts.

Vous pouvez connecter votre application à Internet en utilisant l'une des méthodes suivantes :

- Un sous-réseau public avec une passerelle Internet

  Utilisez des sous-réseaux publics lorsque vous avez des applications publiques qui nécessitent de grandes quantités de bande passante ou une latence minimale. Les scénarios applicables incluent le streaming vidéo et les services de jeux.
- Un sous-réseau privé avec une passerelle NAT

  Utilisez des sous-réseaux privés lorsque vous souhaitez protéger vos conteneurs d'un accès externe direct. Les scénarios applicables incluent les systèmes de traitement des paiements ou les conteneurs stockant les données utilisateur et les mots de passe.
- AWS PrivateLink

  AWS PrivateLink Utilisez-le pour bénéficier d'une connectivité privée entre les VPCs AWS services et vos réseaux locaux sans exposer votre trafic à l'Internet public.

Accès aux fonctionnalités

Vous pouvez utiliser les paramètres de votre compte Amazon ECS pour accéder aux fonctionnalités suivantes :

- Container Insights

  CloudWatch Container Insights collecte, agrège et résume les métriques et les journaux de vos applications conteneurisées et de vos microservices. Les métriques incluent l'utilisation des ressources telles que l'UC, la mémoire, le disque et le réseau.
- `awsvpctronquage`

  Pour certains types d' EC2 instances, des interfaces réseau supplémentaires (ENIs) peuvent être disponibles sur les instances de conteneur récemment lancées.
- Autorisation de balisage

Les utilisateurs doivent disposer d'autorisations pour les actions qui créent une ressource, telles queecsCreateCluster. Si des balises sont spécifiées dans l'action de création de ressources, octroie AWS une autorisation supplémentaire à l'ecs:TagResourceaction afin de vérifier si les utilisateurs ou les rôles sont autorisés à créer des balises.

- Conformité de Fargate à la norme FIPS-140

  Fargate prend en charge la norme Federal Information Processing Standard (FIPS-140), qui spécifie les exigences de sécurité pour les modules cryptographiques qui protègent des informations sensibles. Il s'agit de la norme gouvernementale en vigueur aux États-Unis et au Canada, applicable aux systèmes qui doivent être conformes à la loi sur la gestion de la sécurité des informations fédérales (FISMA) ou au programme fédéral de gestion des risques et des autorisations (FedRAMP).

- Modification de l'heure de fin de la tâche Fargate

  Vous pouvez configurer la période d'attente avant que les tâches Fargate ne soient retirées pour être corrigées.

- VPC à double pile

  Autorisez les tâches à communiquer entre IPv6 elles ou les deux. IPv4

- Format Amazon Resource Name (ARN)

  Certaines fonctionnalités, telles que l'autorisation de balisage, nécessitent un nouveau format Amazon Resource Name (ARN).

Pour de plus amples informations, veuillez consulter Accédez aux fonctionnalités d'Amazon ECS avec les paramètres du compte.

Rôles IAM

Un rôle IAM est une identité IAM que vous pouvez créer dans votre compte et qui dispose d’autorisations spécifiques. Dans Amazon ECS, vous pouvez créer des rôles pour accorder des autorisations à des ressources Amazon ECS telles que des conteneurs ou des services.

Certaines fonctionnalités d'Amazon ECS nécessitent des rôles. Pour de plus amples informations, veuillez consulter Rôles IAM pour Amazon ECS.

Journalisation

La journalisation et la surveillance sont des aspects importants du maintien de la fiabilité, de la disponibilité et des performances des charges de travail Amazon ECS. Les options suivantes sont disponibles :

- Amazon CloudWatch logs - acheminer les journaux vers Amazon CloudWatch
- FireLenspour Amazon ECS : acheminez les journaux vers un AWS service ou une AWS Partner Network destination pour le stockage et l'analyse des journaux. AWS Partner Network Il s'agit d'une communauté mondiale de partenaires qui tire parti des programmes, de l'expertise et des ressources pour créer, commercialiser et vendre des offres aux clients.

Types de lancement Amazon ECS

Le type de lancement de la définition de tâche définit la capacité sur laquelle la tâche peut être exécutée, par exemple AWS Fargate.

Après avoir choisi le type de lancement, Amazon ECS vérifie que les paramètres de définition de tâche que vous configurez fonctionnent avec le type de lancement.

Le tableau suivant décrit les types de lancement disponibles.

Type de lancement		En savoir plus
Fargate	Fargate est un moteur de calcul sans serveur pay-as-you-go qui vous permet de vous concentrer sur la création d'applications sans gérer de serveurs.	Type de lancement Fargate pour Amazon ECS
EC2	Vous choisissez le type d'instance, le nombre d'instances et vous gérez la capacité.	EC2 type de lancement pour Amazon ECS
Machines virtuelles (VM) ou serveurs sur site	Amazon ECS Anywhere fournit une assistance pour l'enregistrement d'une	Type de lancement externe (Amazon ECS Anywhere) pour Amazon ECS

Type de lancement		En savoir plus
	instance externe, telle qu'un serveur sur site ou une machine virtuelle (VM), dans votre cluster.	

Type de lancement Fargate pour Amazon ECS

Fargate est un moteur de calcul sans serveur pay-as-you-go qui vous permet de vous concentrer sur la création d'applications sans gérer de serveurs. Lorsque vous choisissez Fargate, vous n'avez pas besoin de gérer une infrastructure. EC2 Il vous suffit de créer votre image de conteneur et de définir le cluster sur lequel vous souhaitez exécuter vos applications. Fargate propose une intégration AWS native avec des services tels que :

- Amazon VPC
- Auto Scaling
- Elastic Load Balancing
- IAM
- Secrets Manager

Vous avez plus de contrôle avec Fargate EC2 que si vous sélectionnez exactement le processeur et la mémoire dont votre application a besoin. Fargate gère l'augmentation de votre capacité, vous n'avez donc pas à vous soucier des pics de trafic. Cela signifie qu'il y a moins d'efforts opérationnels avec Fargate.

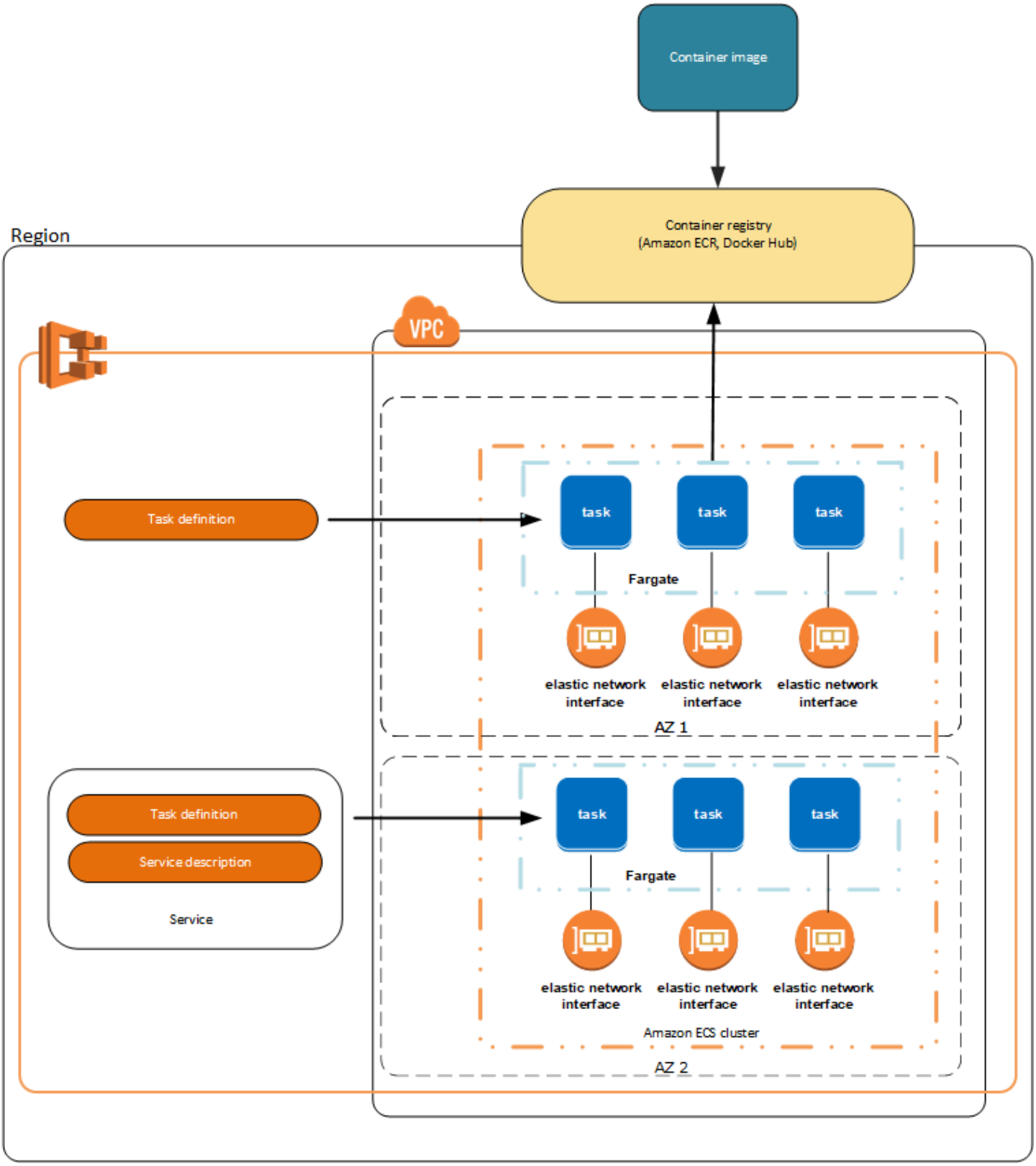
Fargate répond aux normes des programmes de conformité, notamment PCI, FIPS 140-3, FedRAMP et HIPAA. Pour plus d'informations, voir AWS Services concernés par programme de conformité.

Fargate convient aux charges de travail suivantes :

- Charges de travail volumineuses nécessitant de faibles frais opérationnels
- Les petites charges de travail qui présentent des hausses occasionnelles
- Les charges de travail minimes
- Les charges de travail par lot

Pour de plus amples informations sur les Régions qui prennent en charge Fargate, veuillez consulter the section called “AWS Régions de Fargate”.

Le schéma suivant illustre l'architecture générale.



Pour plus d'informations sur Amazon ECS sur Fargate, veuillez consulter [AWS Fargatepour Amazon ECS](#).

Comportement d'extraction d'images de conteneurs Linux sur Fargate pour Amazon ECS

Chaque tâche Fargate s'exécute sur sa propre instance à usage unique et à locataire unique. Lorsque vous exécutez des conteneurs Linux sur Fargate, les images de conteneurs ou les couches d'images de conteneurs ne sont pas mises en cache sur l'instance. Par conséquent, pour chaque image de conteneur définie dans la tâche, l'image de conteneur complète doit être extraite du registre d'images de conteneur pour chaque tâche Fargate. Le temps nécessaire pour extraire les images est directement corrélé au temps nécessaire pour démarrer une tâche Fargate.

Tenez compte des points suivants pour optimiser le temps d'extraction de l'image.

Proximité des images du conteneur

Pour réduire le temps nécessaire au téléchargement des images du conteneur, localisez les données le plus près possible du calcul. Le transfert d'une image de conteneur sur Internet ou sur Internet Régions AWS peut avoir un impact sur le temps de téléchargement. Nous vous recommandons de stocker l'image du conteneur dans la même région que celle où la tâche sera exécutée. Si vous stockez l'image du conteneur dans Amazon ECR, utilisez un point de terminaison d'interface VPC pour réduire davantage le temps d'extraction de l'image. Pour plus d'informations, consultez la section Points de terminaison VPC de l'interface Amazon ECR AWS PrivateLink() dans le guide de l'utilisateur Amazon ECR.

Réduction de la taille de l'image du conteneur

La taille d'une image de conteneur a un impact direct sur le temps de téléchargement. La réduction de la taille de l'image du conteneur ou du nombre de couches de l'image du conteneur peut réduire le temps nécessaire au téléchargement d'une image. Les images de base légères (telles que l'image de conteneur minimale Amazon Linux 2023) peuvent être nettement plus petites que celles basées sur des images de base de système d'exploitation traditionnelles. Pour plus d'informations sur l'image minimale, consultez la section AL2023 Image de conteneur minimale dans le guide de l'utilisateur Amazon Linux 2023.

Algorithmes de compression alternatifs

Les couches d'images de conteneurs sont souvent compressées lorsqu'elles sont transférées vers un registre d'images de conteneurs. La compression de la couche d'image du conteneur réduit la quantité de données qui doivent être transférées sur le réseau et stockées dans le registre des images du conteneur. Une fois qu'une couche d'image de conteneur a été téléchargée sur une instance par le moteur d'exécution du conteneur, cette couche est décompressée. L'algorithme

de compression utilisé et la quantité de v CPUs disponible à l'exécution ont un impact sur le temps nécessaire pour décompresser l'image du conteneur. Sur Fargate, vous pouvez augmenter la taille de la tâche ou tirer parti de l'algorithme de compression zstd, plus performant, pour réduire le temps de décompression. Pour plus d'informations, consultez zstd on. GitHub Pour plus d'informations sur la façon d'implémenter les images pour Fargate, consultez la section AWS Fargate Réduction des temps de démarrage avec les images de conteneur compressées zstd.

Images du conteneur Lazy Loading

Pour les images de conteneur de grande taille (> 250 Mo), il peut être préférable de charger une image de conteneur en différé plutôt que de télécharger l'intégralité de l'image de conteneur. Sur Fargate, vous pouvez utiliser Seekable OCI (SOCI) pour charger en différé une image de conteneur à partir d'un registre d'images de conteneur. Pour plus d'informations, consultez les sections soci-snapshotter on GitHub et Lazy loading container images using Seekable OCI (SOCI).

Comportement d'extraction des images des conteneurs Windows sur Fargate pour Amazon ECS

Fargate Windows met en cache l'image de base du serveur du mois le plus récent et du mois précédent fournie par Microsoft. Ces images correspondent au KB/Build nombre de patchs mis à jour chaque Patch Tuesday. Par exemple, le 9 avril 2024, Microsoft a publié le KB5 036896 (17763.5696) pour Windows Server 2019. Le mois précédent KB le 12 mars 2024 était de KB5 03:5849 (17763.5576). Ainsi, pour les plateformes `WINDOWS_SERVER_2019_CORE` et `WINDOWS_SERVER_2019_FULL` les images de conteneurs suivantes ont été mises en cache :

- `mcr.microsoft.com/windows/servercore:ltsc2019`
- `mcr.microsoft.com/windows/servercore:10.0.17763.5696`
- `mcr.microsoft.com/windows/servercore:10.0.17763.5576`

De plus, le 9 avril 2024, Microsoft a publié le KB5 036909 (20348.2402) pour Windows Server 2022. Le mois précédent KB le 12 mars 2024 était de KB5 03:5857 (20348.2340). Ainsi, pour les plateformes `WINDOWS_SERVER_2022_CORE` et `WINDOWS_SERVER_2022_FULL` les images de conteneurs suivantes ont été mises en cache :

- `mcr.microsoft.com/windows/servercore:ltsc2022`
- `mcr.microsoft.com/windows/servercore:10.0.20348.2402`

- `mcr.microsoft.com/windows/servercore:10.0.20348.2340`

EC2 type de lancement pour Amazon ECS

Le type de EC2 lancement convient aux charges de travail importantes qui doivent être optimisées en termes de prix.

Lorsque vous réfléchissez à la manière de modéliser les définitions de tâches et les services à l'aide du type de EC2 lancement, nous vous recommandons de déterminer quels processus doivent être exécutés ensemble et comment vous pouvez procéder pour dimensionner chaque composant.

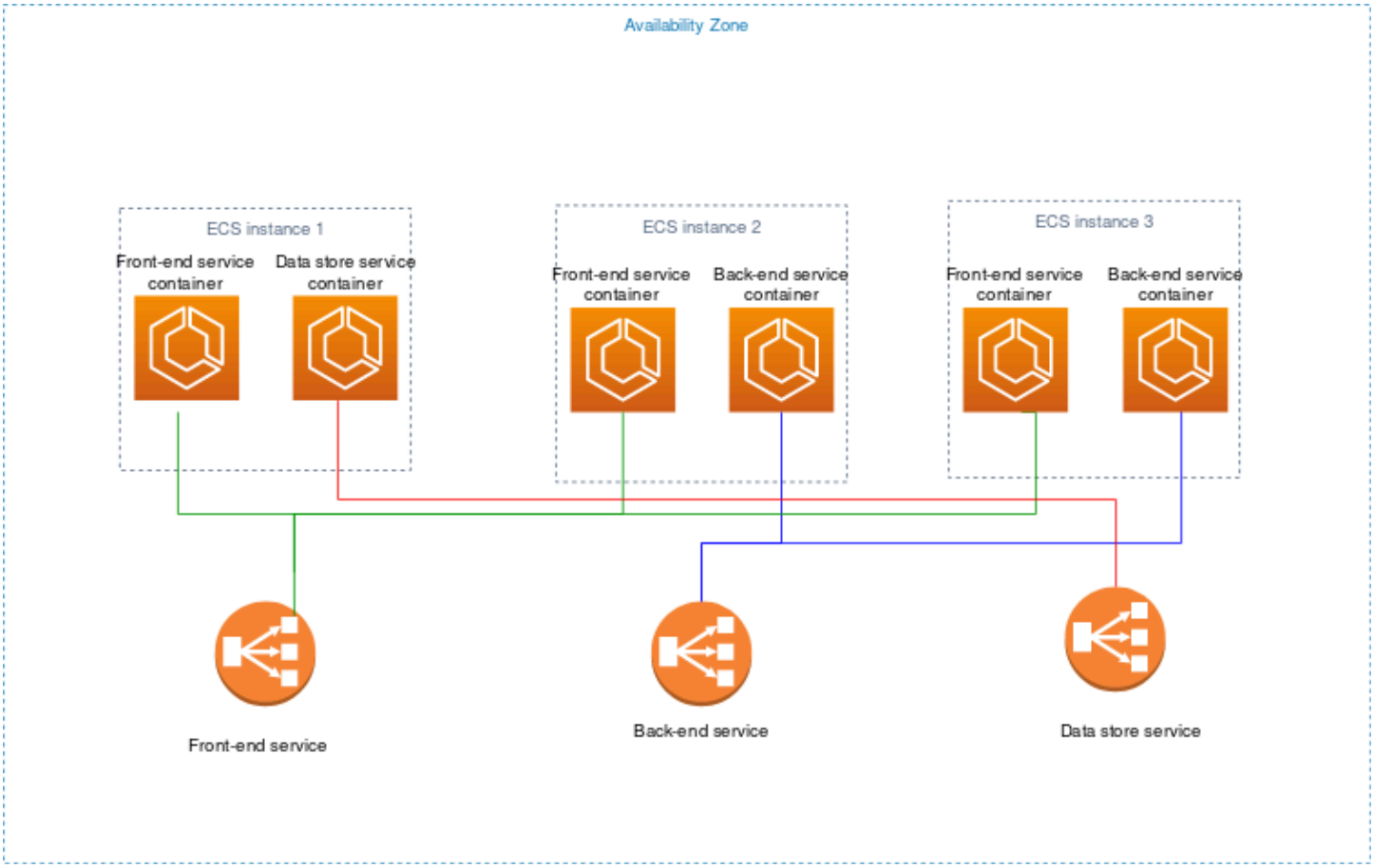
Par exemple, supposons qu'une application comporte les composants suivants :

- Un service frontal qui affiche des informations sur une page web
- Un service backend qui fournit APIs le service frontend
- Une banque de données

Pour cet exemple, créez des définitions de tâches qui regroupent les conteneurs utilisés dans un même but. Séparez les différents composants en plusieurs définitions de tâches distinctes. L'exemple de cluster suivant comporte trois instances de conteneur qui exécutent trois conteneurs de service frontal, deux conteneurs de service backend et un conteneur de services de magasin de données.

Vous pouvez regrouper les conteneurs connexes dans une définition de tâche, comme les conteneurs liés qui doivent être exécutés ensemble. Par exemple, ajoutez un conteneur de flux de journaux à votre service frontal et l'inclure dans la même définition de tâche.

Une fois que vous avez créé vos définitions de tâche, vous pouvez créer des services à partir de ces dernières afin de préserver la disponibilité des tâches souhaitées. Pour de plus amples informations, veuillez consulter Création d'un déploiement de mises à jour continues Amazon ECS. Dans vos services, vous pouvez associer des conteneurs avec des équilibreurs de charge Elastic Load Balancing. Pour de plus amples informations, veuillez consulter Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS. Lorsque les besoins de votre application changent, vous pouvez mettre à jour vos services pour augmenter ou réduire le nombre de tâches souhaitées. Vous pouvez également mettre à jour vos services afin de déployer des versions plus récentes des conteneurs dans vos tâches. Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.



Comportement d'extraction d'images de conteneur pour les types de lancement externe EC2 et externe pour Amazon ECS

Le temps nécessaire au démarrage d'un conteneur varie en fonction de l'image du conteneur sous-jacente. Par exemple, le démarrage d'une image plus volumineuse (versions complètes de Debian, Ubuntu et Amazon1/2) peut prendre plus de temps car les conteneurs contiennent un plus grand nombre de services que leurs versions allégées respectives (Debian-slim, Ubuntu-slim et Amazon-slim) ou les images de base plus petites (Alpine).

Lorsque l'agent Amazon ECS lance une tâche, il extrait l'image Docker de son registre distant, puis met en cache une copie locale. Lorsque vous utilisez une nouvelle balise d'image pour chaque version de votre application, ce comportement n'est pas nécessaire.

Le paramètre de `ECS_IMAGE_PULL_BEHAVIOR` l'agent détermine le comportement d'extraction de l'image. Les options suivantes sont disponibles :

- `ECS_IMAGE_PULL_BEHAVIOR`: `default`

L'image sera extraite à distance. Si l'extraction échoue, l'image mise en cache dans l'instance est utilisée.

- `ECS_IMAGE_PULL_BEHAVIOR`: `always`

  L'image sera extraite à distance. Si le pull échoue, la tâche échoue.

Pour accélérer le déploiement, définissez le paramètre de l'agent Amazon ECS sur l'une des valeurs suivantes :

- `ECS_IMAGE_PULL_BEHAVIOR`: `once`

  L'image est extraite à distance uniquement si elle n'a pas été extraite par une tâche précédente sur la même instance de conteneur ou si l'image mise en cache a été supprimée par le processus de nettoyage automatique de l'image. Dans le cas contraire, l'image mise en cache sur l'instance est utilisée. Cela garantit qu'aucune extraction d'image inutile ne soit tentée.

- `ECS_IMAGE_PULL_BEHAVIOR`: `prefer-cached`

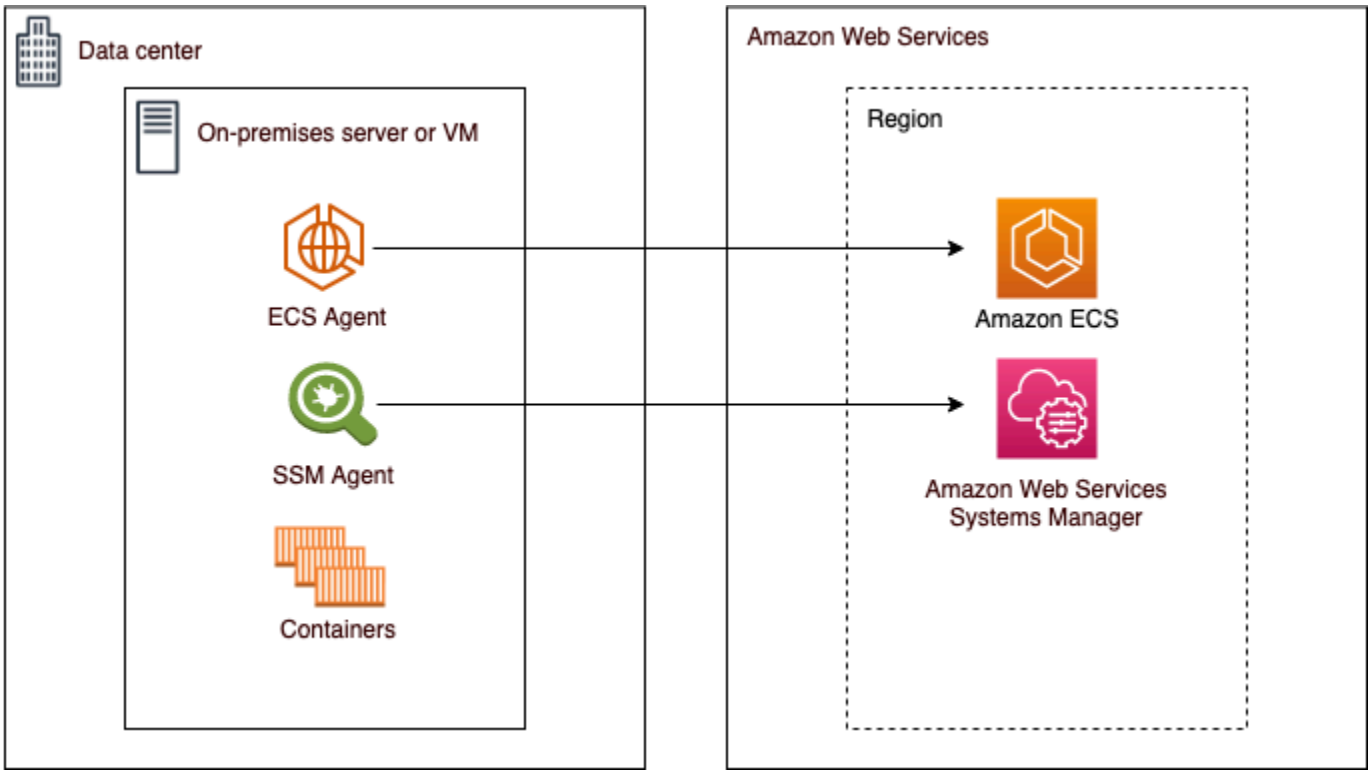
  L'image est extraite à distance s'il n'y a pas d'image mise en cache. Dans le cas contraire, l'image mise en cache sur l'instance est utilisée. Le nettoyage automatique des images est désactivé pour le conteneur afin de garantir que l'image mise en cache n'est pas supprimée.

La définition du `ECS_IMAGE_PULL_BEHAVIOR` paramètre sur l'une des valeurs précédentes permet de gagner du temps, car l'agent Amazon ECS utilise l'image téléchargée existante. Pour les images Docker de plus grande taille, le temps de téléchargement peut prendre de 10 à 20 secondes pour être transférées sur le réseau.

Type de lancement externe (Amazon ECS Anywhere) pour Amazon ECS

Amazon ECS Anywhere fournit un support pour l'enregistrement d'une Instance externe, telle qu'un serveur sur site ou une machine virtuelle (VM), sur votre cluster Amazon ECS. Les instances externes sont optimisées pour exécuter des applications qui génèrent du trafic sortant ou des données de processus. Si votre application nécessite du trafic entrant, l'absence de prise en charge d'Elastic Load Balancing rend l'exécution de ces applications moins efficace. Amazon ECS a ajouté un nouveau type de lancement `EXTERNAL` que vous pouvez utiliser pour créer des services ou exécuter des tâches sur vos instances externes.

Vous trouverez ci-dessous une présentation de l'architecture système de haut niveau d'Amazon ECS Anywhere. L'agent Amazon ECS et l'agent SSM sont installés sur votre serveur sur site.



Pour de plus amples informations, veuillez consulter Clusters Amazon ECS pour le type de lancement externe.

Utilisation des points de terminaison à double pile Amazon ECS

Les points de terminaison à double pile Amazon ECS prennent en charge les demandes adressées à Amazon ECS via le protocole Internet version 4 (IPv4) et le protocole Internet version 6 (IPv6). Pour obtenir la liste de tous les points de terminaison Amazon ECS, consultez la section Points de terminaison et quotas Amazon ECS dans le. Références générales AWS

Lorsque vous utilisez l'API REST, vous accédez directement à un point de terminaison Amazon ECS en utilisant le nom du point de terminaison (URI). Amazon ECS prend uniquement en charge les noms de point de terminaison régionaux à double pile, ce qui signifie que vous devez spécifier la région dans le nom.

Utilisez la convention de dénomination suivante pour les noms de point de terminaison à double pile :`ecs.`*`region`*`.api.aws`.

Lorsque vous utilisez le AWS Command Line Interface (AWS CLI) et AWS SDKs, vous pouvez utiliser un paramètre ou un indicateur pour passer à un point de terminaison à double pile. Vous pouvez également spécifier le point de terminaison à double pile directement en remplacement du point de terminaison Amazon ECS dans le fichier de configuration.

Les sections suivantes décrivent comment utiliser les points de terminaison à double pile à partir de l'API REST AWS CLI, de AWS SDKs, et de l'API REST.

Rubriques

- Utilisation de points de terminaison Dual-Stack avec l AWS CLI
- En utilisant des points de terminaison à double pile provenant du AWS SDKs
- Utilisation de points de terminaison Dual-Stack avec l'API REST

Utilisation de points de terminaison Dual-Stack avec l AWS CLI

Cette section fournit des exemples de AWS CLI commandes utilisées pour envoyer des demandes à un point de terminaison à double pile. Pour plus d'informations sur l'installation AWS CLI ou la mise à jour vers la dernière version, voir Installation ou mise à jour vers la dernière version du AWS CLI Guide de l'AWS Command Line Interface utilisateur de la version 2.

Pour utiliser un point de terminaison à double pile, vous pouvez définir la valeur `use_dualstack_endpoint` de configuration sur `true` dans le `config` fichier AWS CLI afin de diriger toutes les demandes Amazon ECS effectuées par la `ecs` AWS CLI commande vers le point de terminaison à double pile pour la région spécifiée. Vous pouvez spécifier la région dans le `config` fichier ou dans une commande à l'aide de l'`--region`option. Pour plus d'informations sur les fichiers de configuration pour le AWS CLI, consultez la section Paramètres des fichiers de configuration et d'identification AWS CLI dans le guide de l'AWS Command Line Interface utilisateur de la version 2.

Si vous souhaitez utiliser un point de terminaison à double pile pour des AWS CLI commandes spécifiques, vous pouvez utiliser l'une des méthodes suivantes :

- Vous pouvez utiliser le point de terminaison à double pile par commande en définissant le `--endpoint-url` paramètre sur `https://ecs.`*`aws-region`*`.api.aws` ou `http://ecs.`*`aws-region`*`.api.aws` pour n'importe quelle `ecs` commande.

  L'exemple de commande suivant répertorie tous les clusters disponibles et utilise le point de terminaison à double pile pour la demande.

```
$ aws ecs list-clusters --endpoint-url https://ecs.aws-region.api.aws
```

- Vous pouvez configurer des profils distincts dans votre AWS Config fichier. Par exemple, vous pouvez créer un profil qui définit `use_dualstack_endpoint` sur `true` et un profil qui ne définit pas `use_dualstack_endpoint`. Lorsque vous exécutez une commande, spécifiez le profil adéquat selon que vous comptez utiliser ou non le point de terminaison Dual-Stack.

En utilisant des points de terminaison à double pile provenant du AWS SDKs

Cette section fournit des exemples de la manière d'accéder à un point de terminaison à double pile à l'aide du AWS SDKs.

AWS SDK for Java 2.x

L'exemple suivant montre comment spécifier un point de terminaison à double pile pour la `us-east-1` région à l'aide du AWS SDK for Java 2.x.

```
Region region = Region.US_EAST_1
EcsClient client =
 EcsClient.builder().region(region).dualstackEnabled(true).build();
```

AWS SDK pour Go

L'exemple suivant montre comment spécifier un point de terminaison à double pile pour la `us-east-1` région à l'aide du AWS SDK pour Go.

```
sess := session.Must(session.NewSession())
svc := ecs.New(sess, &aws.Config{
    Region: aws.String(endpoints.UsEast1RegionID),
    Endpoint: aws.String("https://ecs.us-east-1.api.aws")
})
```

Pour plus d'informations, consultez la section Points de terminaison à double pile et FIPS dans le guide de référence AWS SDKs et Tools.

Utilisation de points de terminaison Dual-Stack avec l'API REST

Lorsque vous utilisez l'API REST, vous pouvez accéder directement à un point de terminaison à double pile en le spécifiant dans votre demande. L'exemple suivant utilise le point de terminaison à double pile pour répertorier tous les clusters Amazon ECS de la `us-east-1` région.

```
POST / HTTP/1.1
Host: ecs.us-east-1.api.aws
Accept-Encoding: identity
Content-Length: 2
X-Amz-Target: AmazonEC2ContainerServiceV20141113.ListClusters
X-Amz-Date: 20150429T170621Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{}
```

Applications Amazon ECS dans des sous-réseaux partagés, des zones locales et des zones de longueur d'onde

Amazon ECS prend en charge les charges de travail qui utilisent des zones locales, des zones de longueur d'onde, et AWS Outposts lorsqu'une faible latence ou un traitement local des données sont requis.

- Vous pouvez utiliser les Zones Locales comme extension et Région AWS pour placer des ressources dans plusieurs emplacements plus proches de vos utilisateurs finaux.
- Vous pouvez utiliser les zones Wavelength pour créer des applications qui offrent des latences ultra-faibles aux appareils 5G et aux utilisateurs finaux. Wavelength déploie des services de AWS calcul et de stockage standard à la périphérie des réseaux 5G des opérateurs de télécommunications.
- AWS Outposts intègre des modèles natifs Services AWS, d'infrastructure et d'exploitation à pratiquement tous les centres de données, espaces de colocation ou installations sur site.

> **Important**
>
> Amazon ECS sur les AWS Fargate charges de travail n'est pas pris en charge dans les zones Local, les zones de longueur d'onde ou autres pour le AWS Outposts moment.

Pour plus d'informations sur les différences entre les zones locales, les zones de longueur d'onde et AWS Outposts , voir How should I think about when to use AWS Wavelength, AWS Local Zones, or AWS Outposts for applications nécessitant une faible latence ou un traitement local des données dans le AWS Wavelength FAQs.

Sous-réseaux partagés

Vous pouvez utiliser Partage de VPC pour partager des sous-réseaux avec d'autres comptes AWS au sein d'un même AWS Organizations.

Vous pouvez utiliser le mode partagé VPCs pour le type de EC2 lancement en tenant compte des considérations suivantes :

- Le propriétaire du sous-réseau VPC doit partager un sous-réseau avec un compte participant avant que ce compte puisse l'utiliser pour les ressources Amazon ECS.
- Vous ne pouvez pas utiliser le groupe de sécurité VPC par défaut pour vos instances de conteneur car il appartient au propriétaire. En outre, les participants ne peuvent pas lancer d'instances à l'aide de groupes de sécurité appartenant à d'autres participants ou au propriétaire.
- Dans un sous-réseau partagé, le participant et le propriétaire contrôlent séparément les groupes de sécurité au sein de leur compte respectif. Le propriétaire du sous-réseau peut voir ces groupes de sécurité créés par les participants, mais ne peut pas exécuter d'actions sur ceux-ci. Si le propriétaire du sous-réseau souhaite supprimer ou modifier ces groupes de sécurité, le participant qui a créé le groupe de sécurité doit effectuer l'action.
- Le propriétaire du VPC partagé ne peut pas afficher, mettre à jour ou supprimer un cluster créé par un participant dans le sous-réseau partagé. Cela s'ajoute aux ressources VPC auxquelles chaque compte a un accès différent. Pour plus d'informations, consultez Responsabilités et autorisations pour les propriétaires et les participants dans le Guide de l'utilisateur Amazon VPC.

Vous pouvez utiliser le mode partagé VPCs pour le type de lancement Fargate en tenant compte des considérations suivantes :

- Le propriétaire du sous-réseau VPC doit partager un sous-réseau avec un compte participant avant que ce compte puisse l'utiliser pour les ressources Amazon ECS.
- Vous ne pouvez pas créer de service ou exécuter une tâche à l'aide du groupe de sécurité par défaut du VPC, car celui-ci appartient au propriétaire. En outre, les participants ne peuvent pas créer de service ou exécuter une tâche à l'aide de groupes de sécurité appartenant à d'autres participants ou au propriétaire.
- Dans un sous-réseau partagé, le participant et le propriétaire contrôlent séparément les groupes de sécurité au sein de leur compte respectif. Le propriétaire du sous-réseau peut voir ces groupes de sécurité créés par les participants, mais ne peut pas exécuter d'actions sur ceux-ci. Si le propriétaire du sous-réseau souhaite supprimer ou modifier ces groupes de sécurité, le participant qui a créé le groupe de sécurité doit effectuer l'action.
- Le propriétaire du VPC partagé ne peut pas afficher, mettre à jour ou supprimer un cluster créé par un participant dans le sous-réseau partagé. Cela s'ajoute aux ressources VPC auxquelles chaque compte a un accès différent. Pour plus d'informations, consultez Responsabilités et autorisations pour les propriétaires et les participants dans le Guide de l'utilisateur Amazon VPC.

Pour plus d'informations sur le partage de sous-réseaux VPC, consultez Partager votre VPC avec d'autres comptes dans le Guide de l'utilisateur Amazon VPC.

Zones locales

Une zone locale est une extension d'une Région AWS zone géographique très proche de vos utilisateurs. Les Local Zones ont leurs propres connexions à Internet et prennent en charge AWS Direct Connect. Les ressources créées dans une zone locale peuvent servir les utilisateurs locaux avec des communications à faible latence. Pour plus d'informations, consultez Local Zones AWS.

Une zone locale est représentée par un code de région suivi d'un identifiant qui indique l'emplacement (par exemple, `us-west-2-lax-1a`).

Pour utiliser une Local Zone, vous devez d'abord vous inscrire à la zone. Une fois que vous avez choisi, vous devez créer un VPC Amazon et un sous-réseau dans la Local Zone.

Vous pouvez lancer des EC2 instances Amazon, des serveurs de FSx fichiers Amazon et des équilibreurs de charge d'application à utiliser pour vos clusters et tâches Amazon ECS.

Pour plus d'informations, voir Qu'est-ce que AWS les zones locales ? dans le Guide de l'utilisateur des Zones AWS Locales.

Zones Wavelength

Vous pouvez utiliser AWS Wavelength pour créer des applications qui offrent une latence ultra-faible aux appareils mobiles et aux utilisateurs finaux. Wavelength déploie des services de AWS calcul et de stockage standard à la périphérie des réseaux 5G des opérateurs de télécommunications. Vous pouvez étendre un Amazon Virtual Private Cloud à une ou plusieurs zones Wavelength. Vous pouvez ensuite utiliser AWS des ressources telles que des EC2 instances Amazon pour exécuter des applications nécessitant une latence très faible et une connexion Services AWS à la région.

Une zone Wavelength est une zone isolée située à l'emplacement du transporteur où l'infrastructure Wavelength est déployée. Les zones de longueur d'onde sont liées à un Région AWS. Une zone Wavelength est une extension logique d'une région et est gérée par le plan de contrôle de la région.

Une zone Wavelength est représentée par un code de région suivi d'un identifiant qui indique la zone Wavelength (par exemple, `us-east-1-wl1-bos-wlz-1`).

Pour utiliser une zone Wavelength, vous devez d'abord vous inscrire à la zone. Une fois que vous avez choisi, vous devez créer un VPC Amazon et un sous-réseau dans la zone Wavelength. Vous pouvez ensuite lancer vos EC2 instances Amazon dans la zone afin de les utiliser pour vos clusters et tâches Amazon ECS.

Pour plus d'informations, consultez Mise en route avec AWS Wavelength dans le Guide du développeur AWS Wavelength .

Les zones de longueur d'onde ne sont pas toutes disponibles Régions AWS. Pour plus d'informations sur les régions qui prennent en charge les zones Wavelength, consultez Zones Wavelength disponibles dans le Guide du développeur AWS Wavelength .

Amazon Elastic Container Service sur AWS Outposts

AWS Outposts permet AWS des services, une infrastructure et des modèles d'exploitation natifs dans les installations sur site. Dans AWS Outposts les environnements, vous pouvez utiliser les mêmes AWS APIs outils et infrastructures que ceux que vous utilisez dans le AWS Cloud.

Amazon ECS on AWS Outposts est idéal pour les charges de travail à faible latence qui doivent être exécutées à proximité de données et d'applications sur site.

Pour plus d'informations AWS Outposts, consultez le guide de AWS Outposts l'utilisateur.

Considérations

Voici les points à prendre en compte lors de l'utilisation d'Amazon ECS sur AWS Outposts :

- Amazon Elastic Container Registry et Network Load Balancer s'exécutent dans la AWS région, et non sur. AWS Identity and Access Management AWS Outposts Cela a pour effet d'accroître les latences entre ces services et les conteneurs.
- AWS Fargate n'est pas disponible sur AWS Outposts.

Voici les considérations relatives à la connectivité réseau pour AWS Outposts :

- Si la connectivité réseau entre votre AWS région AWS Outposts et sa région est perdue, vos clusters continueront à fonctionner. Toutefois, vous ne pouvez pas créer de nouveaux clusters ni effectuer de nouvelles actions sur les clusters existants tant que la connectivité n'a pas été rétablie. En cas de défaillance d'instance, l'instance ne sera pas automatiquement remplacée. L'agent CloudWatch Logs ne sera pas en mesure de mettre à jour les journaux et les données des événements.
- Nous vous recommandons de fournir une connectivité fiable, à haute disponibilité et à faible latence entre votre AWS région AWS Outposts et la sienne.

Prérequis

Les conditions requises pour utiliser Amazon ECS sont les suivantes : AWS Outposts

- Vous devez avoir installé et configuré un Outpost dans votre centre de données sur site.
- Vous devez avoir une connexion réseau fiable entre votre Outpost et sa région AWS .

Vue d'ensemble de la création de clusters sur AWS Outposts

Voici un aperçu de la configuration :

1. Créez un rôle et une politique avec des droits activés AWS Outposts.
2. Créez un profil d'instance IAM avec des droits sur AWS Outposts.
3. Créez un VPC ou utilisez-en un existant situé dans la même région que le vôtre. AWS Outposts
4. Créez un sous-réseau ou utilisez un sous-réseau existant associé au AWS Outposts.

Il s'agit du sous-réseau sur lequel s'exécutent les instances de conteneur.

5. Créez un groupe de sécurité pour les instances de conteneur de votre cluster.
6. Créez un nouveau cluster Amazon ECS.
7. Définissez les variables d'environnement de l'agent de conteneur Amazon ECS pour lancer l'instance dans le cluster.
8. Faites fonctionner un conteneur.

Pour des informations détaillées sur la façon d'intégrer Amazon ECS à Amazon ECS AWS Outposts, consultez Étendre Amazon ECS sur deux AWS Outposts racks.

L'exemple suivant crée un cluster Amazon ECS sur un AWS Outposts.

1. Créez un rôle et une politique avec des droits activés AWS Outposts.

   Le fichier `role-policy.json` est le document de politique qui contient l'effet et les actions des ressources. Pour plus d'informations sur le format de fichier, consultez le manuel PutRolePolicyde référence de l'API IAM

```
aws iam create-role --role-name ecsRole \
    --assume-role-policy-document file://ecs-policy.json
aws iam put-role-policy --role-name ecsRole --policy-name ecsRolePolicy \
    --policy-document file://role-policy.json
```

2. Créez un profil d'instance IAM avec des droits sur AWS Outposts.

```
aws iam create-instance-profile --instance-profile-name outpost
aws iam add-role-to-instance-profile --instance-profile-name outpost \
    --role-name ecsRole
```

3. Créez un VPC.

```
aws ec2 create-vpc --cidr-block 10.0.0.0/16
```

4. Créez un sous-réseau associé à votre. AWS Outposts

```
aws ec2 create-subnet \
    --cidr-block 10.0.3.0/24 \
    --vpc-id vpc-xxxxxxxx \
```

```
    --outpost-arn arn:aws:outposts:us-west-2:123456789012:outpost/op-
xxxxxxxxxxxxxxxx \
    --availability-zone-id usw2-az1
```

5. Créez un groupe de sécurité pour les instances de conteneur, en spécifiant la plage CIDR appropriée pour AWS Outposts. (Cette étape est différente pour AWS Outposts.)

```
aws ec2 create-security-group --group-name MyOutpostSG
aws ec2 authorize-security-group-ingress --group-name MyOutpostSG --protocol tcp \
    --port 22 --cidr 10.0.3.0/24
aws ec2 authorize-security-group-ingress --group-name MyOutpostSG --protocol tcp \
    --port 80 --cidr 10.0.3.0/24
```

6. Créez le cluster.
7. Définissez les variables d'environnement de l'agent de conteneur Amazon ECS pour lancer l'instance dans le cluster créé à l'étape précédente et définissez les identifications que vous souhaitez ajouter afin d'identifier le cluster (par exemple `Outpost` pour indiquer que le cluster est destiné à un Outpost).

```
#! /bin/bash
cat << 'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_IMAGE_PULL_BEHAVIOR=prefer-cached
ECS_CONTAINER_INSTANCE_TAGS={"environment": "Outpost"}
EOF
```

> **Note**
>
> Pour éviter les retards causés par l'extraction d'images de conteneur d'Amazon ECR dans la région, utilisez des caches d'images. Pour ce faire, chaque fois qu'une tâche est exécutée, configurez l'agent Amazon ECS pour qu'il utilise par défaut l'image mise en cache sur l'instance elle-même en définissant `ECS_IMAGE_PULL_BEHAVIOR` sur `prefer-cached`.

8. Créez l'instance de conteneur, en spécifiant le VPC et le sous-réseau de l' AWS Outposts sur lequel cette instance doit s'exécuter et un type d'instance disponible sur l' AWS Outposts. (Cette étape est différente pour AWS Outposts.)

   Le fichier `userdata.txt` contient les données des utilisateurs que l'instance peut utiliser pour effectuer des tâches de configuration automatisées courantes et même exécuter des scripts

après le démarrage de l'instance. Pour plus d'informations sur le fichier pour les appels d'API, consultez la section Exécuter des commandes sur votre instance Linux au lancement dans le guide de EC2 l'utilisateur Amazon.

```
aws ec2 run-instances --count 1 --image-id ami-xxxxxxxx --instance-type c5.large \
    --key-name aws-outpost-key --subnet-id subnet-xxxxxxxxxxxxxxxxx \
    --iam-instance-profile Name outpost --security-group-id sg-xxxxxx \
    --associate-public-ip-address --user-data file://userdata.txt
```

> **Note**
>
> Cette commande sert également à ajouter des instances supplémentaires au cluster. Tous les conteneurs déployés dans le cluster sont placés sur cet AWS Outposts.

Bonnes pratiques de mise en réseau Amazon ECS

Les applications modernes sont généralement construites à partir de plusieurs composants distribués qui communiquent entre eux. Par exemple, une application mobile ou Web peut communiquer avec un point de terminaison d'API, et l'API peut être alimentée par plusieurs microservices qui communiquent via Internet.

Pour plus d'informations sur les meilleures pratiques relatives aux applications de connexion à Internet, consultezConnect les applications Amazon ECS à Internet.

Pour plus d'informations sur les meilleures pratiques relatives à la réception de connexions entrantes vers Amazon ECS depuis Internet, consultezBonnes pratiques pour recevoir des connexions entrantes vers Amazon ECS depuis Internet.

Pour plus d'informations sur les meilleures pratiques pour connecter Amazon ECS à d'autres AWS services, consultezBonnes pratiques pour connecter Amazon ECS aux AWS services depuis votre VPC.

Pour plus d'informations sur les meilleures pratiques en matière de connexion de services au sein d'un VPC, consultez. Meilleures pratiques pour connecter les services Amazon ECS dans un VPC

Pour plus d'informations sur les meilleures pratiques en matière de services réseau entre Comptes AWS et VPCs, consultezBonnes pratiques pour la mise en réseau des services Amazon ECS entre Comptes AWS et VPCs.

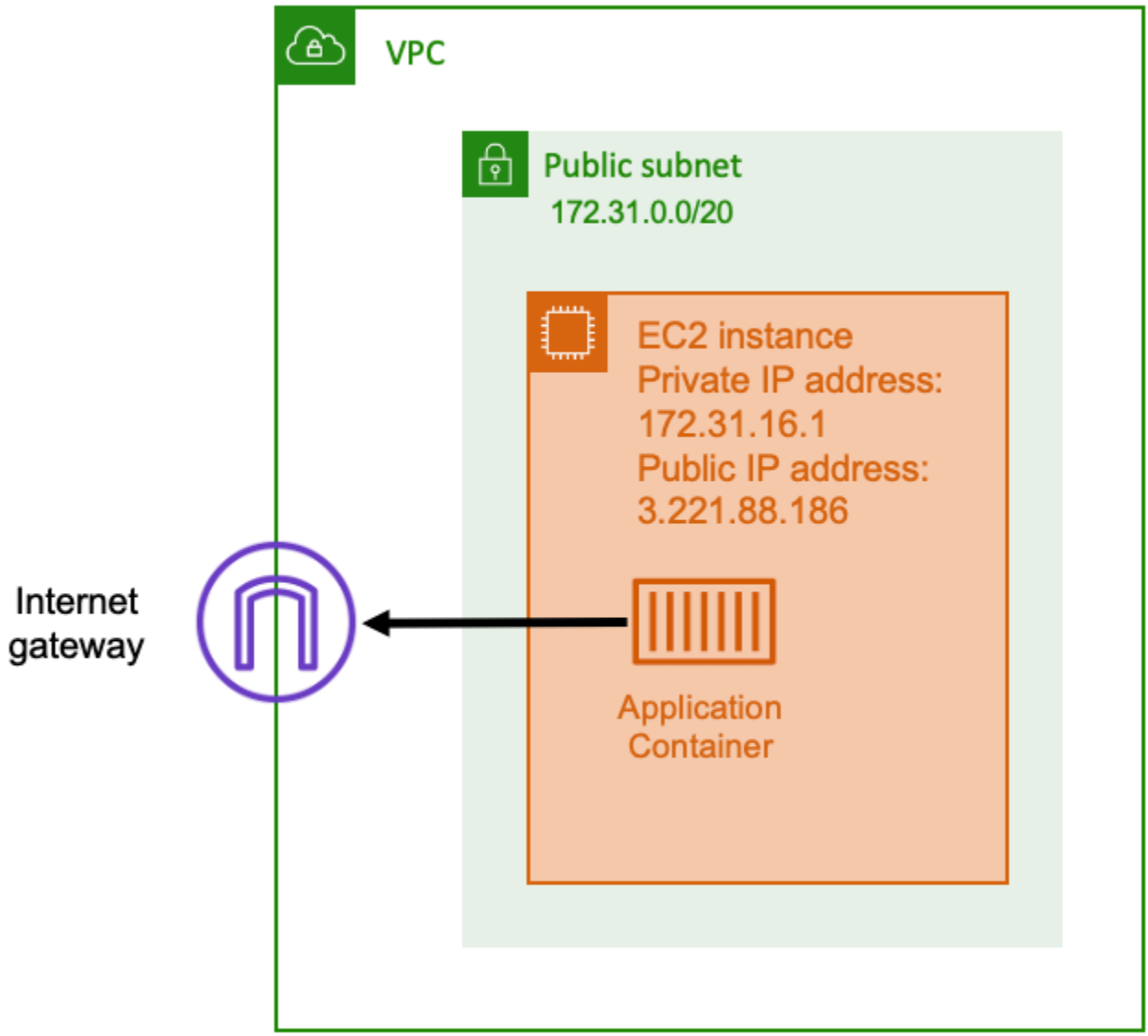
Pour plus d'informations sur les meilleures pratiques relatives aux services permettant de résoudre les problèmes de réseau, consultez. AWS services de dépannage réseau Amazon ECS

Connect les applications Amazon ECS à Internet

La plupart des applications conteneurisées comportent au moins certains composants qui nécessitent un accès sortant à Internet. Par exemple, le backend d'une application mobile nécessite un accès sortant aux notifications push.

Amazon Virtual Private Cloud propose deux méthodes principales pour faciliter la communication entre votre VPC et Internet.

Sous-réseau public et passerelle Internet



Lorsque vous utilisez un sous-réseau public doté d'une route vers une passerelle Internet, votre application conteneurisée peut s'exécuter sur un hôte au sein d'un VPC situé sur un sous-réseau public. Une adresse IP publique est attribuée à l'hôte qui gère votre conteneur. Cette adresse IP publique est routable depuis Internet. Pour plus d'informations, consultez Passerelles Internet dans le Guide de l'utilisateur Amazon VPC.

Cette architecture réseau facilite la communication directe entre l'hôte qui exécute votre application et les autres hôtes sur Internet. La communication est bidirectionnelle. Cela signifie que non seulement vous pouvez établir une connexion sortante avec n'importe quel autre hôte sur Internet, mais que d'autres hôtes sur Internet peuvent également tenter de se connecter à votre hôte. Par conséquent,

vous devez porter une attention particulière à votre groupe de sécurité et à vos règles de pare-feu. Cela garantit que les autres hôtes sur Internet ne peuvent pas ouvrir de connexions que vous ne souhaitez pas voir ouvertes.

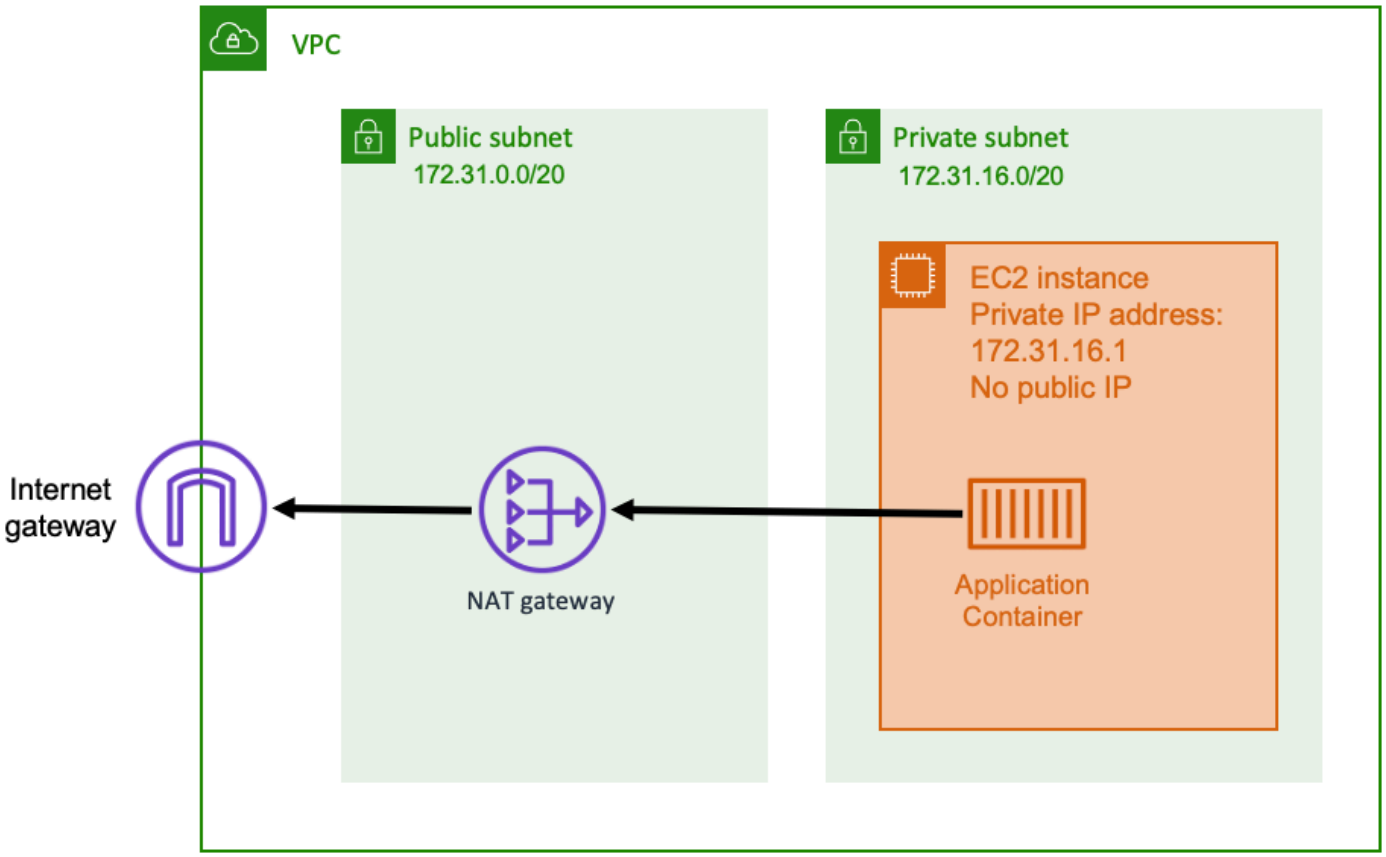
Par exemple, si votre application s'exécute sur Amazon EC2, assurez-vous que le port 22 pour l'accès SSH n'est pas ouvert. Dans le cas contraire, votre instance pourrait recevoir des tentatives de connexion SSH constantes de la part de robots malveillants sur Internet. Ces robots parcourent les adresses IP publiques. Une fois qu'ils ont trouvé un port SSH ouvert, ils tentent d'utiliser des mots de passe par force brute pour essayer d'accéder à votre instance. De ce fait, de nombreuses entreprises limitent l'utilisation des sous-réseaux publics et préfèrent que la plupart, sinon la totalité, de leurs ressources se trouvent dans des sous-réseaux privés.

L'utilisation de sous-réseaux publics pour la mise en réseau convient aux applications publiques qui nécessitent de grandes quantités de bande passante ou une latence minimale. Les cas d'utilisation applicables incluent le streaming vidéo et les services de jeux.

Cette approche réseau est prise en charge à la fois lorsque vous utilisez Amazon ECS sur Amazon EC2 et lorsque vous l'utilisez sur AWS Fargate.

- Amazon EC2 — Vous pouvez lancer EC2 des instances sur un sous-réseau public. Amazon ECS utilise ces EC2 instances comme capacité de cluster, et tous les conteneurs exécutés sur les instances peuvent utiliser l'adresse IP publique sous-jacente de l'hôte pour le réseau sortant. Cela s'applique à la fois au mode `bridge` réseau `host` et au mode réseau. Cependant, le mode `awsvpc` réseau ne fournit pas d'adresses IP publiques ENIs à la tâche. Ils ne peuvent donc pas utiliser directement une passerelle Internet.
- Fargate — Lorsque vous créez votre service Amazon ECS, spécifiez des sous-réseaux publics pour la configuration réseau de votre service et utilisez l'option Attribuer une adresse IP publique. Chaque tâche Fargate est mise en réseau dans le sous-réseau public et possède sa propre adresse IP publique pour une communication directe avec Internet.

Sous-réseau privé et passerelle NAT



Lorsque vous utilisez un sous-réseau privé et une passerelle NAT, vous pouvez exécuter votre application conteneurisée sur un hôte situé dans un sous-réseau privé. Ainsi, cet hôte possède une adresse IP privée qui est routable au sein de votre VPC, mais qui n'est pas routable depuis Internet. Cela signifie que les autres hôtes du VPC peuvent se connecter à l'hôte à l'aide de son adresse IP privée, mais que les autres hôtes sur Internet ne peuvent pas établir de communications entrantes avec l'hôte.

Avec un sous-réseau privé, vous pouvez utiliser une passerelle de traduction d'adresses réseau (NAT) pour permettre à un hôte au sein d'un sous-réseau privé de se connecter à Internet. Les hôtes sur Internet reçoivent une connexion entrante qui semble provenir de l'adresse IP publique de la passerelle NAT située dans un sous-réseau public. La passerelle NAT est chargée de servir de pont entre Internet et le sous-réseau privé. Cette configuration est souvent préférée pour des raisons de sécurité, car elle signifie que votre VPC est protégé contre tout accès direct par des attaquants sur Internet. Pour plus d'informations, veuillez consulter NAT Gateways (Passerelles NAT) dans le Guide de l'utilisateur Amazon VPC.

Cette approche de réseau privé convient aux scénarios dans lesquels vous souhaitez protéger vos conteneurs d'un accès externe direct. Les scénarios applicables incluent les systèmes de traitement des paiements ou les conteneurs stockant les données utilisateur et les mots de passe. La création et l'utilisation d'une passerelle NAT vous sont facturées dans votre compte. L'utilisation horaire de la passerelle NAT et les taux de traitement des données s'appliquent également. Pour des raisons de redondance, vous devez disposer d'une passerelle NAT dans chaque zone de disponibilité. Ainsi, la perte de disponibilité d'une seule zone de disponibilité ne compromet pas votre connectivité sortante. De ce fait, si votre charge de travail est faible, il peut être plus rentable d'utiliser des sous-réseaux privés et des passerelles NAT.

Cette approche réseau est prise en charge à la fois lors de l'utilisation d'Amazon ECS sur Amazon EC2 et lors de son utilisation sur AWS Fargate.

- Amazon EC2 — Vous pouvez lancer EC2 des instances sur un sous-réseau privé. Les conteneurs qui s'exécutent sur ces EC2 hôtes utilisent le réseau des hôtes sous-jacents, et les demandes sortantes passent par la passerelle NAT.
- Fargate — Lorsque vous créez votre service Amazon ECS, spécifiez des sous-réseaux privés pour la configuration réseau de votre service et n'utilisez pas l'option Attribuer une adresse IP publique. Chaque tâche Fargate est hébergée dans un sous-réseau privé. Son trafic sortant est acheminé via n'importe quelle passerelle NAT que vous avez associée à ce sous-réseau privé.

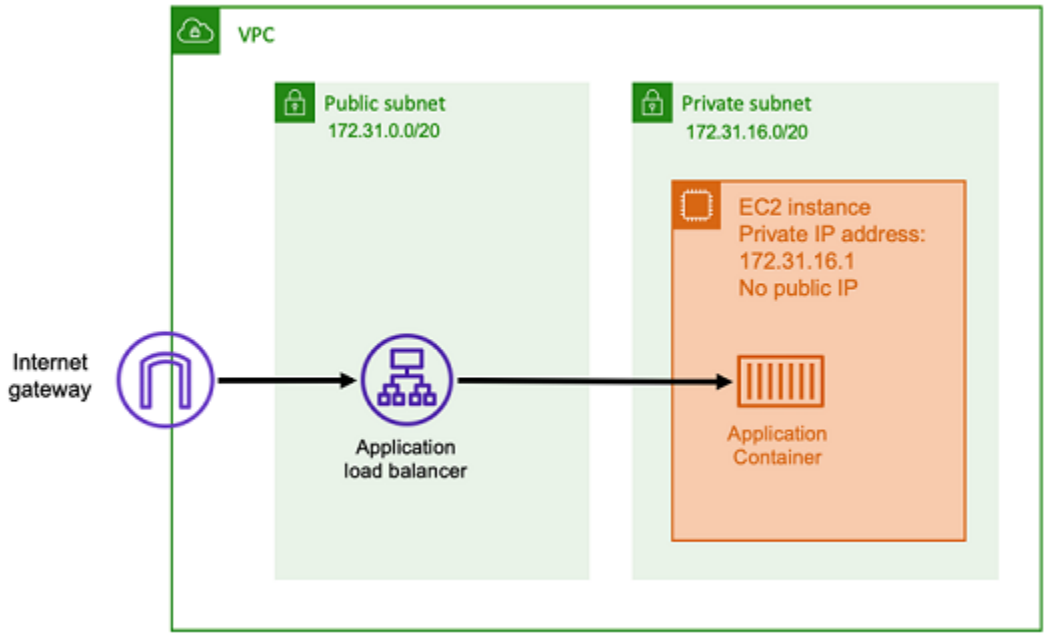
Bonnes pratiques pour recevoir des connexions entrantes vers Amazon ECS depuis Internet

Si vous gérez un service public, vous devez accepter le trafic entrant en provenance d'Internet. Par exemple, votre site Web public doit accepter les requêtes HTTP entrantes provenant des navigateurs. Dans ce cas, les autres hôtes sur Internet doivent également établir une connexion entrante avec l'hôte de votre application.

Une solution à ce problème consiste à lancer vos conteneurs sur des hôtes situés dans un sous-réseau public doté d'une adresse IP publique. Toutefois, nous ne le recommandons pas pour les applications à grande échelle. Pour ces derniers, une meilleure approche consiste à disposer d'une couche d'entrée évolutive située entre Internet et votre application. Pour cette approche, vous pouvez utiliser n'importe lequel des AWS services répertoriés dans cette section comme entrée.

Application Load Balancer

Un Application Load Balancer fonctionne au niveau de la couche application. Il s'agit de la septième couche du modèle d'interconnexion des systèmes ouverts (OSI). Cela rend un Application Load Balancer adapté aux services HTTP publics. Si vous avez un site Web ou une API REST HTTP, un Application Load Balancer est un équilibreur de charge adapté à cette charge de travail. Pour plus d'informations, voir Qu'est-ce qu'un Application Load Balancer ? dans le guide de l'utilisateur des équilibreurs de charge d'application.



Avec cette architecture, vous créez un Application Load Balancer dans un sous-réseau public afin qu'il dispose d'une adresse IP publique et puisse recevoir des connexions entrantes depuis Internet. Lorsque l'Application Load Balancer reçoit une connexion entrante, ou plus précisément une requête HTTP, il ouvre une connexion à l'application à l'aide de son adresse IP privée. Ensuite, il transmet la demande via la connexion interne.

Un Application Load Balancer présente les avantages suivants.

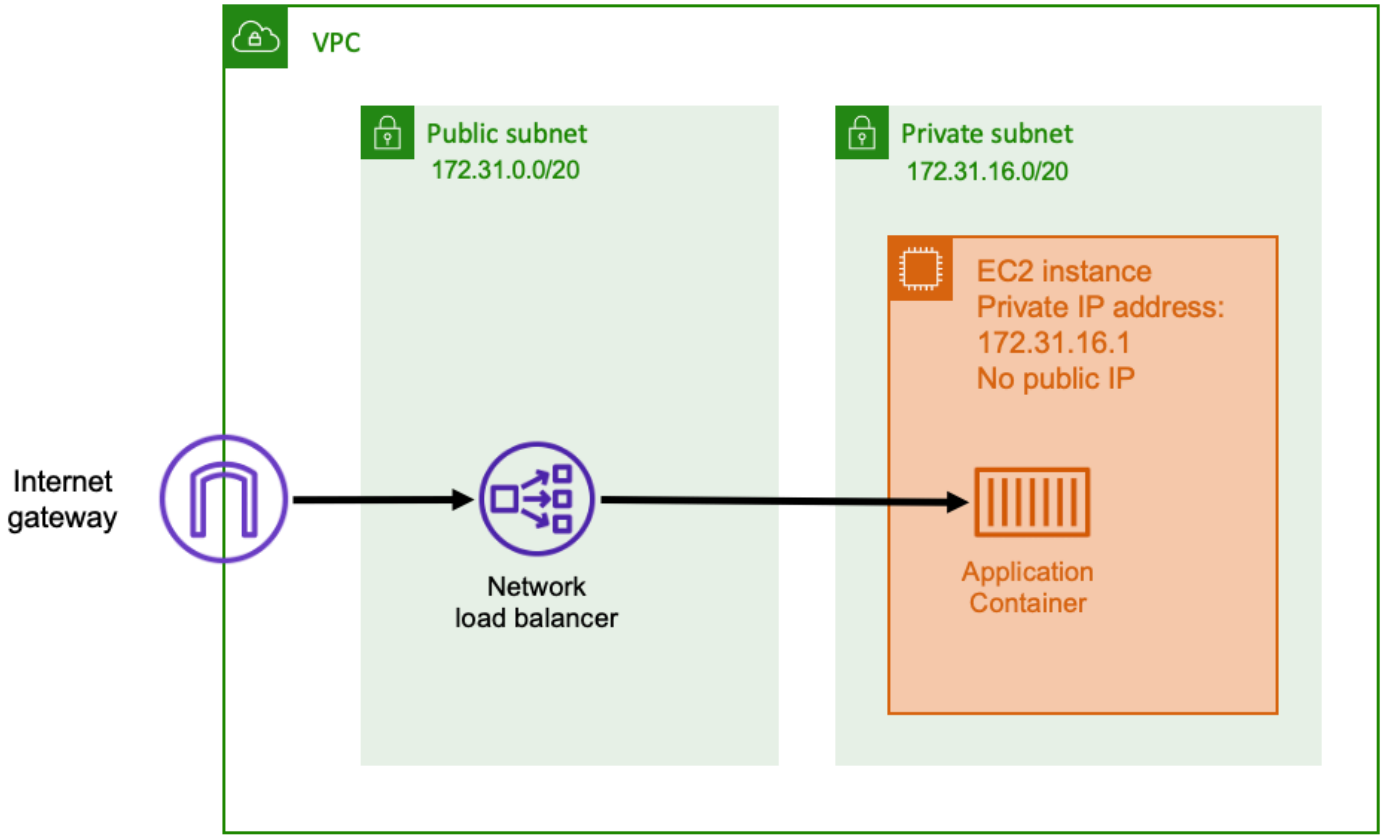
- Résiliation SSL/TLS : un Application Load Balancer peut garantir une communication HTTPS sécurisée et des certificats pour les communications avec les clients. Il peut éventuellement mettre fin à la connexion SSL au niveau de l'équilibreur de charge afin que vous n'ayez pas à gérer les certificats dans votre propre application.
- Routage avancé — Un Application Load Balancer peut avoir plusieurs noms d'hôte DNS. Il dispose également de fonctionnalités de routage avancées pour envoyer des requêtes HTTP entrantes

vers différentes destinations en fonction de métriques telles que le nom d'hôte ou le chemin de la demande. Cela signifie que vous pouvez utiliser une seule Application Load Balancer comme entrée pour de nombreux services internes différents, voire des microservices sur différents chemins d'une API REST.

- Support du gRPC et websockets — Un Application Load Balancer peut gérer bien plus que du simple HTTP. Il peut également équilibrer la charge des services basés sur le gRPC et le websocket, avec le support HTTP/2.
- Sécurité — Un Application Load Balancer permet de protéger votre application contre le trafic malveillant. Il inclut des fonctionnalités telles que l'atténuation de la synchronisation HTTP et est intégré au AWS Web Application Firewall (AWS WAF). AWS WAF peut filtrer davantage le trafic malveillant susceptible de contenir des modèles d'attaque, tels que l'injection SQL ou les scripts intersites.

Network Load Balancer

Un Network Load Balancer fonctionne à la quatrième couche du modèle Open Systems Interconnection (OSI). Il convient aux protocoles non HTTP ou aux scénarios dans lesquels le end-to-end chiffrement est nécessaire, mais il ne possède pas les mêmes fonctionnalités spécifiques au protocole HTTP qu'un Application Load Balancer. Par conséquent, un Network Load Balancer convient parfaitement aux applications qui n'utilisent pas le protocole HTTP. Pour plus d'informations, voir Qu'est-ce qu'un Network Load Balancer ? dans le guide de l'utilisateur pour les équilibreurs de charge réseau.



Lorsqu'un Network Load Balancer est utilisé comme entrée, il fonctionne de la même manière qu'un Application Load Balancer. Cela est dû au fait qu'il est créé dans un sous-réseau public et possède une adresse IP publique accessible sur Internet. Le Network Load Balancer ouvre ensuite une connexion à l'adresse IP privée de l'hôte exécutant votre conteneur et envoie les paquets du côté public au côté privé.

Fonctionnalités du Network Load Balancer

Étant donné que le Network Load Balancer fonctionne à un niveau inférieur de la pile réseau, il ne possède pas le même ensemble de fonctionnalités que l'Application Load Balancer. Cependant, il présente les caractéristiques importantes suivantes.

- End-to-end chiffrement : étant donné qu'un Network Load Balancer fonctionne au niveau de la quatrième couche du modèle OSI, il ne lit pas le contenu des paquets. Cela le rend adapté à l'équilibrage de charge des communications nécessitant un end-to-end chiffrement.
- Chiffrement TLS — Outre le end-to-end chiffrement, Network Load Balancer peut également mettre fin aux connexions TLS. Ainsi, vos applications principales n'ont pas à implémenter leur propre protocole TLS.

- Support UDP : étant donné qu'un Network Load Balancer fonctionne au niveau de la quatrième couche du modèle OSI, il convient aux charges de travail non HTTP et aux protocoles autres que TCP.

Fermeture des connexions

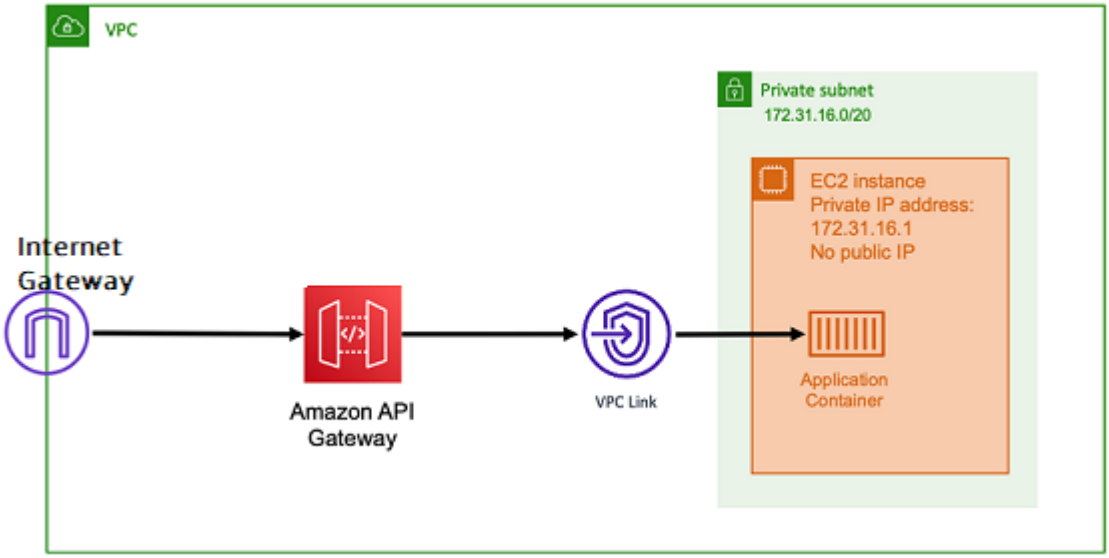
Comme le Network Load Balancer ne respecte pas le protocole d'application dans les couches supérieures du modèle OSI, il ne peut pas envoyer de messages de fermeture aux clients utilisant ces protocoles. Contrairement à l'Application Load Balancer, ces connexions doivent être fermées par l'application ou vous pouvez configurer le Network Load Balancer pour fermer les connexions de quatrième couche lorsqu'une tâche est arrêtée ou remplacée. Consultez le paramètre de terminaison de connexion pour les groupes cibles de Network Load Balancer dans la documentation de Network Load Balancer.

Si le Network Load Balancer ferme les connexions au niveau de la quatrième couche, les clients peuvent afficher des messages d'erreur indésirables s'ils ne les gèrent pas. Pour plus d'infirmations sur la configuration client recommandée, consultez la bibliothèque Builders ici.

Les méthodes pour fermer les connexions varient selon les applications, mais l'une d'entre elles consiste à s'assurer que le délai de désenregistrement cible du Network Load Balancer est plus long que le délai d'expiration de la connexion client. Le client devait d'abord expirer le délai imparti et se reconnecter progressivement à la tâche suivante via le Network Load Balancer, tandis que l'ancienne tâche épuisait lentement tous ses clients. Pour plus d'informations sur le délai de désenregistrement cible du Network Load Balancer, consultez la documentation du Network Load Balancer.

API HTTP Amazon API Gateway

Amazon API Gateway convient aux applications HTTP présentant des pics soudains de volumes de requêtes ou de faibles volumes de requêtes. Pour plus d'informations, consultez Qu'est-ce qu'Amazon API Gateway ? dans le guide du développeur d'API Gateway.



Le modèle de tarification pour Application Load Balancer et Network Load Balancer inclut un prix horaire afin de maintenir les équilibreurs de charge disponibles pour accepter les connexions entrantes à tout moment. En revanche, API Gateway facture chaque demande séparément. Cela a pour effet que, si aucune demande n'est reçue, il n'y a aucun frais. En cas de forte charge de trafic, un Application Load Balancer ou un Network Load Balancer peut traiter un plus grand volume de demandes à un prix par requête inférieur à celui d'API Gateway. Toutefois, si vous avez un faible nombre de demandes dans l'ensemble ou si vous avez des périodes de faible trafic, le prix cumulé de l'utilisation de l'API Gateway devrait être plus rentable que le paiement d'une redevance horaire pour maintenir un équilibreur de charge sous-utilisé. L'API Gateway peut également mettre en cache les réponses de l'API, ce qui peut entraîner une baisse des taux de demandes du backend.

Les fonctions API Gateway utilisent un lien VPC qui permet au service AWS géré de se connecter aux hôtes du sous-réseau privé de votre VPC à l'aide de son adresse IP privée. Il peut détecter ces adresses IP privées en consultant les enregistrements de découverte de AWS Cloud Map services gérés par Amazon ECS Service Discovery.

API Gateway prend en charge les fonctionnalités suivantes.

- Le fonctionnement d'API Gateway est similaire à un équilibreur de charge, mais possède des fonctionnalités supplémentaires propres à la gestion des API.
- L'API Gateway fournit des fonctionnalités supplémentaires concernant l'autorisation des clients, les niveaux d'utilisation et les request/response modifications. Pour plus d'informations, consultez les fonctionnalités d'Amazon API Gateway.

- L'API Gateway peut prendre en charge les points de terminaison de passerelle d'API périphériques, régionaux et privés. Les points de terminaison Edge sont disponibles via une CloudFront distribution gérée. Les points de terminaison régionaux et privés sont tous deux locaux d'une région.
- Terminaison SSL/TLS
- Routage de différents chemins HTTP vers différents microservices principaux

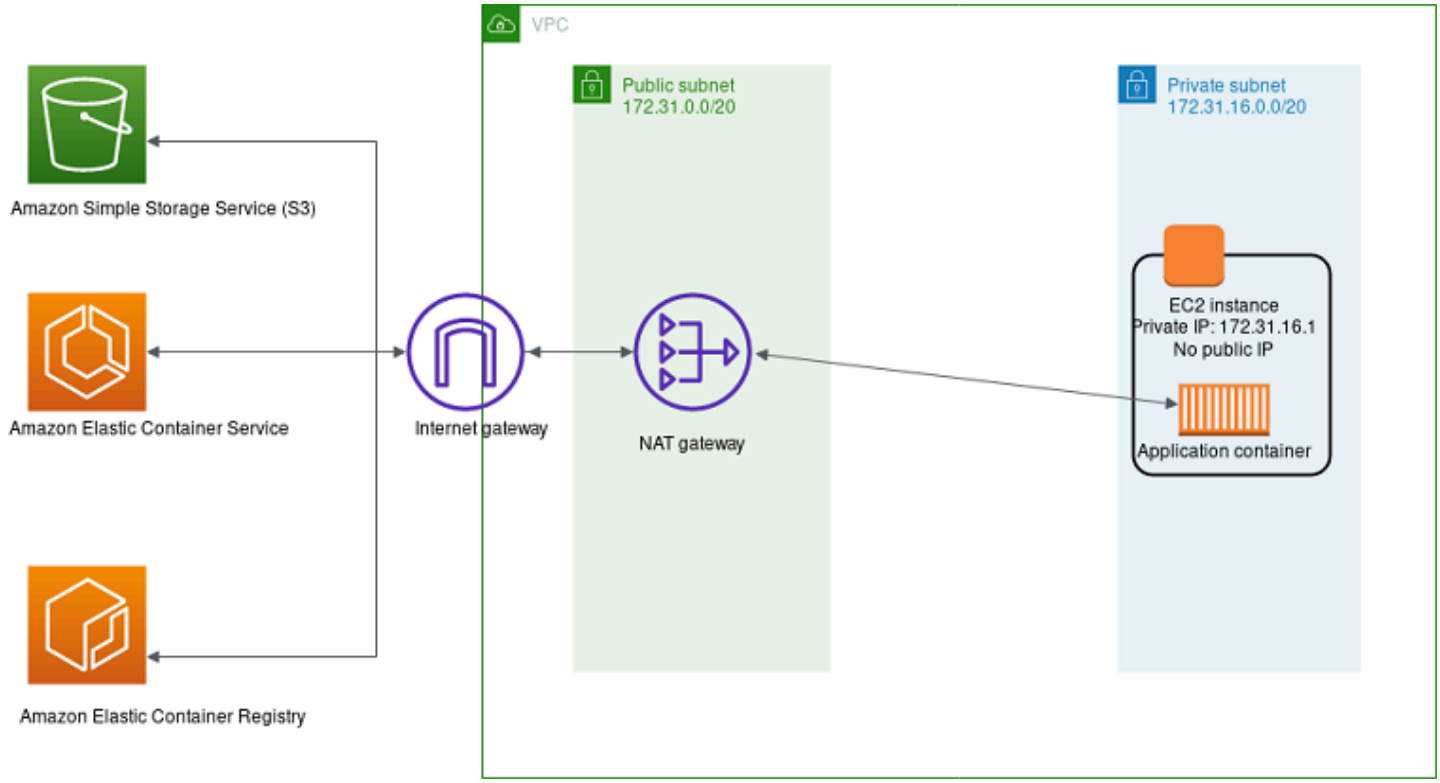
Outre les fonctionnalités précédentes, API Gateway prend également en charge l'utilisation d'autorisateurs Lambda personnalisés que vous pouvez utiliser pour protéger votre API contre toute utilisation non autorisée. Pour plus d'informations, consultez Field Notes : basé sur un conteneur sans serveur avec APIs Amazon ECS et Amazon API Gateway.

Bonnes pratiques pour connecter Amazon ECS aux AWS services depuis votre VPC

Pour qu'Amazon ECS fonctionne correctement, l'agent de conteneur Amazon ECS qui s'exécute sur chaque hôte doit communiquer avec le plan de contrôle Amazon ECS. Si vous stockez vos images de conteneur dans Amazon ECR, les EC2 hôtes Amazon doivent communiquer avec le point de terminaison du service Amazon ECR et avec Amazon S3, où les couches d'images sont stockées. Si vous utilisez d'autres AWS services pour votre application conteneurisée, tels que les données persistantes stockées dans DynamoDB, vérifiez que ces services disposent également de la prise en charge réseau nécessaire.

Passerelle NAT

L'utilisation d'une passerelle NAT est le moyen le plus simple de garantir que vos tâches Amazon ECS peuvent accéder à d'autres AWS services. Pour plus d'informations sur cette approche, consultezSous-réseau privé et passerelle NAT.



Les inconvénients de cette approche sont les suivants :

- Vous ne pouvez pas limiter les destinations avec lesquelles la passerelle NAT peut communiquer. Vous ne pouvez pas non plus limiter les destinations vers lesquelles votre niveau principal peut communiquer sans interrompre toutes les communications sortantes de votre VPC.
- Les passerelles NAT facturent chaque Go de données qui les transite. Si vous utilisez la passerelle NAT pour l'une des opérations suivantes, chaque Go de bande passante vous est facturé :
  - Téléchargement de fichiers volumineux depuis Amazon S3
  - Exécution d'un volume élevé de requêtes de base de données vers DynamoDB
  - Extraction d'images depuis Amazon ECR

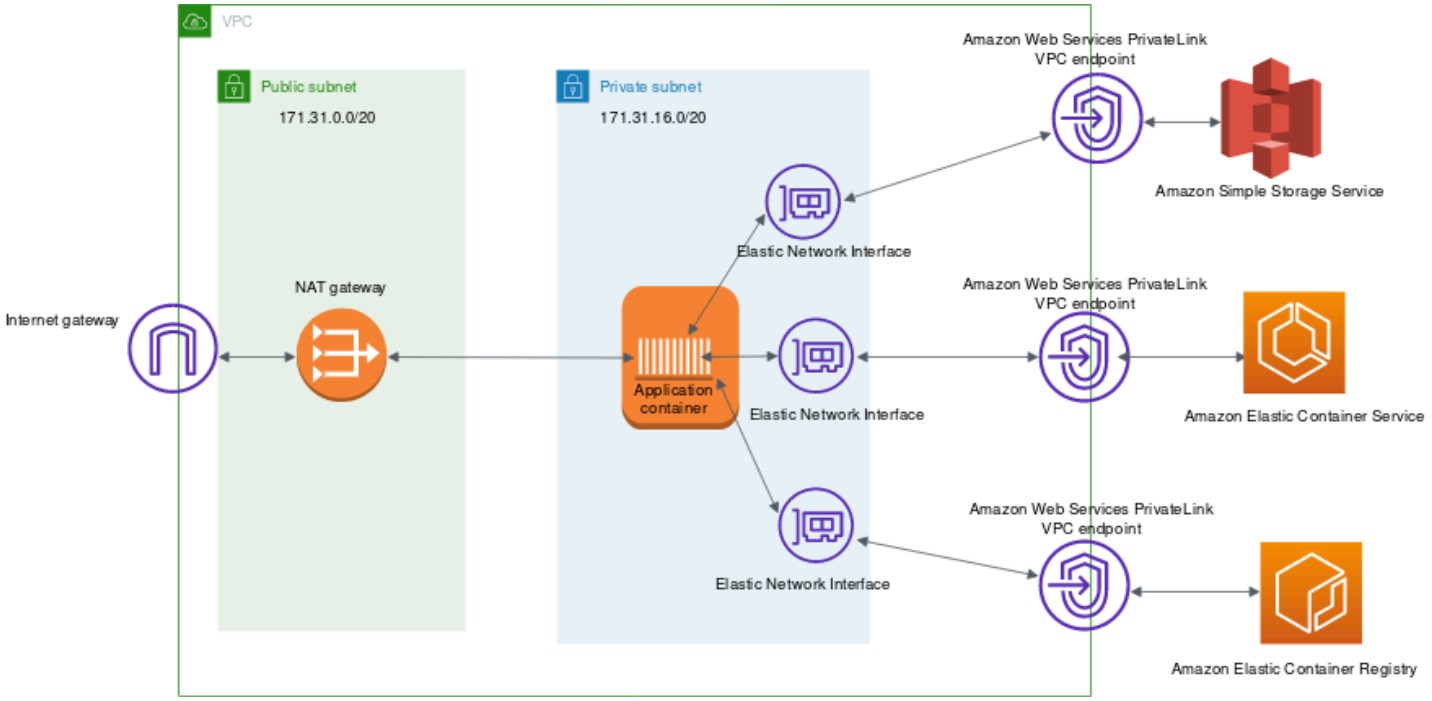
  De plus, les passerelles NAT prennent en charge 5 Gbit/s de bande passante et s'adaptent automatiquement à 45 Gbit/s. Si vous effectuez un routage via une seule passerelle NAT, les applications qui nécessitent des connexions à très haut débit peuvent se heurter à des contraintes réseau. Pour contourner le problème, vous pouvez répartir votre charge de travail sur plusieurs sous-réseaux et attribuer à chaque sous-réseau sa propre passerelle NAT.

AWS PrivateLink

AWS PrivateLink fournit une connectivité privée entre VPCs les AWS services et vos réseaux locaux sans exposer votre trafic à l'Internet public.

Un point de terminaison VPC permet des connexions privées entre votre VPC et les services pris en charge et les services AWS de point de terminaison VPC. Le trafic entre votre VPC et les autres services ne quitte pas le réseau Amazon. Un point de terminaison VPC ne nécessite pas de passerelle Internet, de passerelle privée virtuelle, de périphérique NAT, de connexion VPN ou AWS Direct Connect de connexion. EC2Les instances Amazon de votre VPC n'ont pas besoin d'adresses IP publiques pour communiquer avec les ressources du service.

Le schéma suivant montre comment fonctionne la communication avec les AWS services lorsque vous utilisez des points de terminaison VPC au lieu d'une passerelle Internet. AWS PrivateLink fournit des interfaces réseau élastiques (ENIs) à l'intérieur du sous-réseau, et les règles de routage VPC sont utilisées pour envoyer toute communication au nom d'hôte du service via l'ENI, directement au service de destination. AWS Ce trafic n'a plus besoin d'utiliser la passerelle NAT ou la passerelle Internet.



Voici quelques-uns des points de terminaison VPC courants utilisés avec le service Amazon ECS.

- Points de terminaison de passerelle pour Amazon S3
- Point de terminaison VPC DynamoDB

- Point de terminaison VPC Amazon ECS
- Point de terminaison VPC Amazon ECR

De nombreux autres AWS services prennent en charge les points de terminaison VPC. Si vous utilisez intensivement un AWS service, vous devez consulter la documentation spécifique à ce service et savoir comment créer un point de terminaison VPC pour ce trafic.

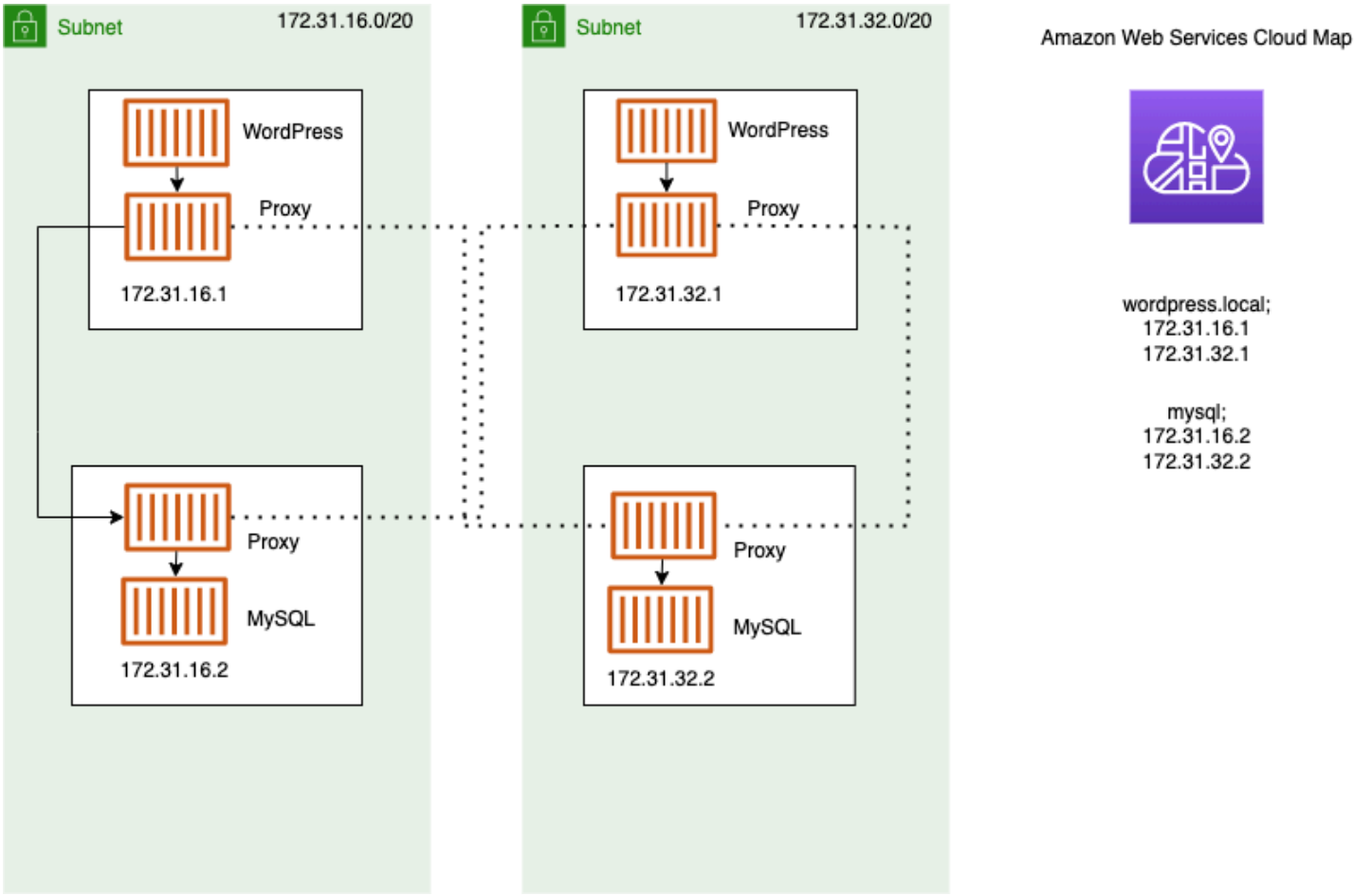
Meilleures pratiques pour connecter les services Amazon ECS dans un VPC

À l'aide des tâches Amazon ECS dans un VPC, vous pouvez diviser les applications monolithiques en parties distinctes qui peuvent être déployées et mises à l'échelle indépendamment dans un environnement sécurisé. Cette architecture est appelée architecture orientée services (SOA) ou microservices. Cependant, il peut être difficile de s'assurer que toutes ces parties, à la fois à l'intérieur et à l'extérieur d'un VPC, peuvent communiquer entre elles. Il existe plusieurs approches pour faciliter la communication, qui présentent toutes des avantages et des inconvénients différents.

Utilisation de Service Connect

Nous recommandons Service Connect, qui fournit une configuration Amazon ECS pour la découverte de services, la connectivité et la surveillance du trafic. Avec Service Connect, vos applications peuvent utiliser des noms abrégés et des ports standard pour se connecter aux services du même cluster ou d'autres clusters, y compris VPCs au sein de la même région. Pour plus d'informations, consultez Amazon ECS Service Connect.

Lorsque vous utilisez Service Connect, Amazon ECS gère toutes les étapes de la découverte de services : création des noms susceptibles d'être découverts, gestion dynamique des entrées pour chaque tâche au démarrage et à l'arrêt des tâches, exécution d'un agent dans chaque tâche configuré pour découvrir les noms. Votre application peut rechercher les noms en utilisant les fonctionnalités standard pour les noms DNS et en établissant des connexions. Si votre application le fait déjà, vous n'avez pas besoin de la modifier pour utiliser Service Connect.



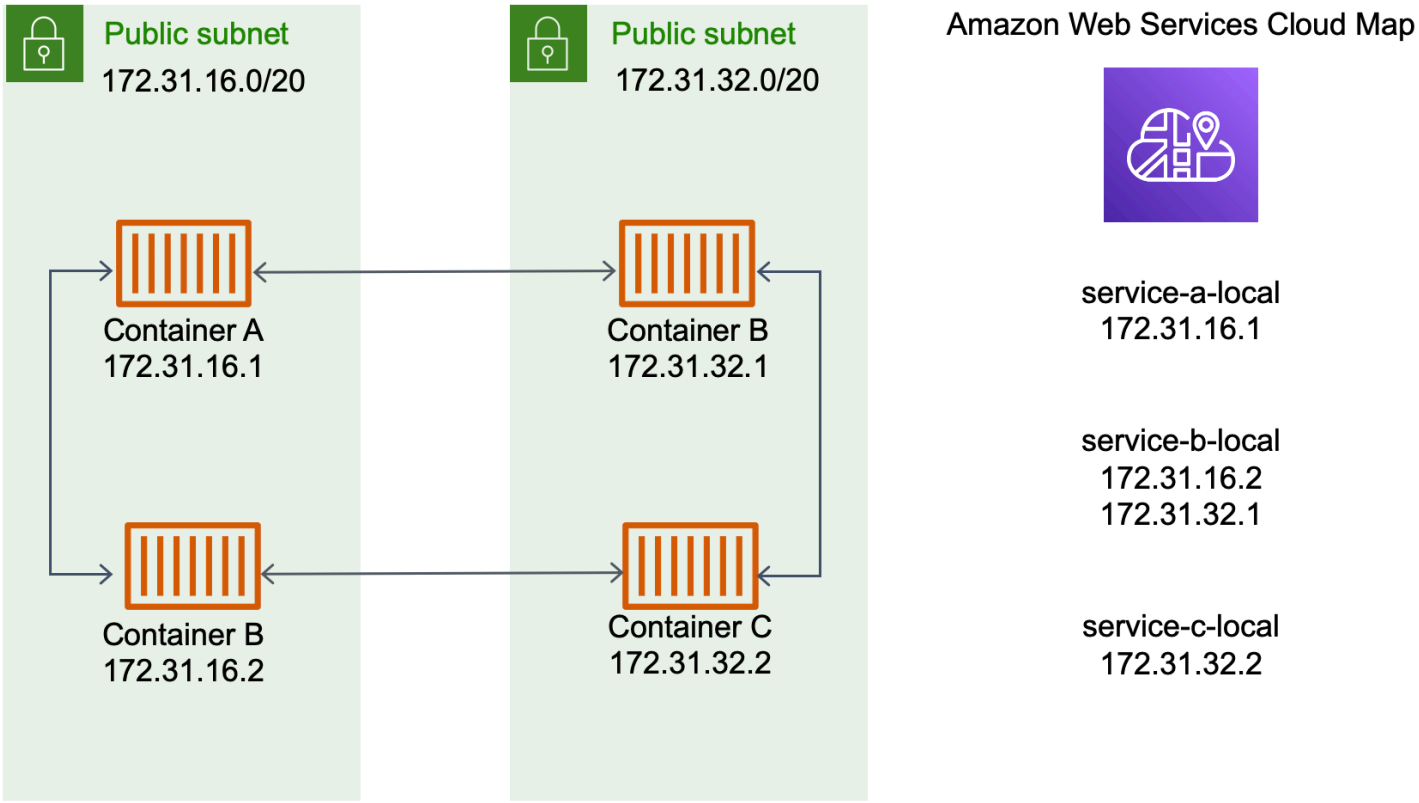
Les modifications ne se produisent que lors des déploiements

Vous fournissez la configuration complète dans chaque définition de service et de tâche. Amazon ECS gère les modifications apportées à cette configuration lors de chaque déploiement de service, afin de garantir que toutes les tâches d'un déploiement se comportent de la même manière. Par exemple, un problème courant lié à la découverte du DNS en tant que service est le contrôle d'une migration. Si vous modifiez un nom DNS pour qu'il pointe vers les nouvelles adresses IP de remplacement, le délai TTL maximal peut être nécessaire avant que tous les clients commencent à utiliser le nouveau service. Avec Service Connect, le déploiement du client met à jour la configuration en remplaçant les tâches du client. Vous pouvez configurer le disjoncteur de déploiement et toute autre configuration de déploiement pour affecter les modifications de Service Connect de la même manière que pour tout autre déploiement.

Utilisation de la découverte de services

Une autre approche de service-to-service communication est la communication directe à l'aide de la découverte de services. Dans cette approche, vous pouvez utiliser l'intégration de la découverte de

AWS Cloud Map services avec Amazon ECS. À l'aide de la découverte de services, Amazon ECS synchronise la liste des tâches lancées avec AWS Cloud Map, qui conserve un nom d'hôte DNS qui correspond aux adresses IP internes d'une ou de plusieurs tâches provenant de ce service en particulier. Les autres services d'Amazon VPC peuvent utiliser ce nom d'hôte DNS pour envoyer du trafic directement vers un autre conteneur à l'aide de son adresse IP interne. Pour de plus amples informations, consultez Découverte de service.



Dans le schéma précédent, il existe trois services. `service-a-local`possède un conteneur et communique avec`service-b-local`, qui possède deux conteneurs. `service-b-local`doit également communiquer avec`service-c-local`, qui possède un conteneur. Chaque conteneur de ces trois services peut utiliser les noms DNS internes de AWS Cloud Map pour rechercher les adresses IP internes d'un conteneur auprès du service en aval avec lequel il doit communiquer.

Cette approche de service-to-service communication permet une faible latence. À première vue, c'est également simple car il n'y a aucun composant supplémentaire entre les conteneurs. Le trafic passe directement d'un conteneur à l'autre.

Cette approche convient lorsque vous utilisez le mode `awsvpc` réseau, où chaque tâche possède sa propre adresse IP unique. La plupart des logiciels ne prennent en charge que l'utilisation d'Aenregistrements DNS, qui se résolvent directement en adresses IP. Lorsque vous utilisez le mode

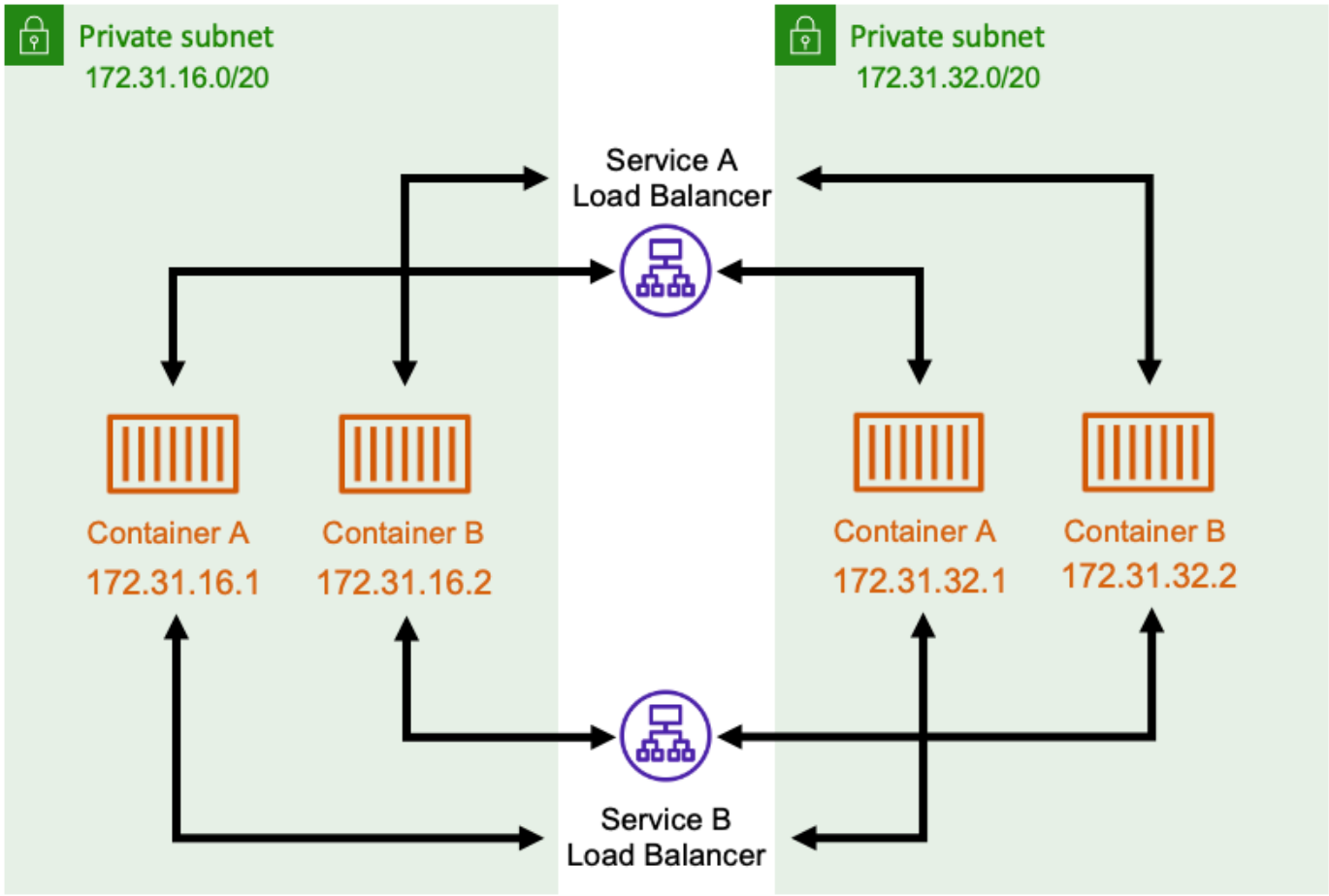
`awsvpc` réseau, l'adresse IP de chaque tâche est un `A` enregistrement. Toutefois, si vous utilisez le mode `bridge` réseau, plusieurs conteneurs peuvent partager la même adresse IP. En outre, les mappages de ports dynamiques entraînent l'attribution aléatoire de numéros de port aux conteneurs sur cette adresse IP unique. À ce stade, un `A` enregistrement ne suffit plus pour la découverte de services. Vous devez également utiliser un `SRV` enregistrement. Ce type d'enregistrement permet de suivre à la fois les adresses IP et les numéros de port, mais nécessite que vous configuriez les applications de manière appropriée. Certaines applications prédéfinies que vous utilisez peuvent ne pas prendre en charge `SRV` les enregistrements.

Un autre avantage du mode `awsvpc` réseau est que vous disposez d'un groupe de sécurité unique pour chaque service. Vous pouvez configurer ce groupe de sécurité pour autoriser les connexions entrantes provenant uniquement des services en amont spécifiques qui doivent communiquer avec ce service.

Le principal inconvénient de la service-to-service communication directe utilisant la découverte de services est que vous devez implémenter une logique supplémentaire pour effectuer de nouvelles tentatives et faire face aux échecs de connexion. Les enregistrements DNS ont une période time-to-live (TTL) qui contrôle la durée pendant laquelle ils sont mis en cache. Il faut un certain temps pour que l'enregistrement DNS soit mis à jour et que le cache expire afin que vos applications puissent récupérer la dernière version de l'enregistrement DNS. Il se peut donc que votre application finisse par résoudre l'enregistrement DNS pour qu'il pointe vers un autre conteneur qui n'existe plus. Votre application doit gérer les nouvelles tentatives et disposer d'une logique lui permettant d'ignorer les mauvais backends.

Utilisation d'un équilibreur de charge interne

Une autre approche de service-to-service communication consiste à utiliser un équilibreur de charge interne. Un équilibreur de charge interne existe entièrement à l'intérieur de votre VPC et n'est accessible qu'aux services situés à l'intérieur de votre VPC.



L'équilibreur de charge maintient une haute disponibilité en déployant des ressources redondantes dans chaque sous-réseau. Lorsqu'un conteneur `serviceA` doit communiquer avec un conteneur depuis`serviceB`, il ouvre une connexion avec l'équilibreur de charge. L'équilibreur de charge ouvre ensuite une connexion vers un conteneur à partir de`service B`. L'équilibreur de charge sert de lieu centralisé pour gérer toutes les connexions entre chaque service.

Si un conteneur s'`service`Barrête, l'équilibreur de charge peut le retirer du pool. L'équilibreur de charge vérifie également l'état de santé de chaque cible en aval de son pool et peut automatiquement supprimer les mauvaises cibles du pool jusqu'à ce qu'elles redeviennent saines. Les applications n'ont plus besoin de connaître le nombre de conteneurs en aval. Ils ouvrent simplement leurs connexions à l'équilibreur de charge.

Cette approche est avantageuse pour tous les modes de réseau. L'équilibreur de charge peut suivre les adresses IP des tâches en mode `awsvpc` réseau, ainsi que les combinaisons plus avancées d'adresses IP et de ports en mode `bridge` réseau. Il répartit le trafic de manière uniforme entre

toutes les combinaisons d'adresses IP et de ports, même si plusieurs conteneurs sont hébergés sur la même EC2 instance Amazon, mais uniquement sur des ports différents.

Le seul inconvénient de cette approche est son coût. Pour être hautement disponible, l'équilibreur de charge doit disposer de ressources dans chaque zone de disponibilité. Cela entraîne des coûts supplémentaires en raison des frais supplémentaires liés au paiement de l'équilibreur de charge et à la quantité de trafic qui passe par l'équilibreur de charge.

Cependant, vous pouvez réduire les frais généraux en faisant en sorte que plusieurs services partagent un équilibreur de charge. Cela convient particulièrement aux services REST qui utilisent un Application Load Balancer. Vous pouvez créer des règles de routage basées sur les chemins qui acheminent le trafic vers différents services. Par exemple, `/api/user/*` peut être acheminé vers un conteneur faisant partie du `user` service, alors qu'il `/api/order/*` peut être acheminé vers le `order` service associé. Avec cette approche, vous ne payez que pour un Application Load Balancer et vous disposez d'une URL cohérente pour votre API. Cependant, vous pouvez répartir le trafic entre différents microservices sur le backend.

Bonnes pratiques pour la mise en réseau des services Amazon ECS entre Comptes AWS et VPCs

Si vous faites partie d'une organisation composée de plusieurs équipes et divisions, vous déployez probablement des services de manière indépendante VPCs dans un environnement partagé Compte AWS ou dans VPCs des services associés à plusieurs personnes Comptes AWS. Quelle que soit la manière dont vous déployez vos services, nous vous recommandons de compléter vos composants réseau pour faciliter l'acheminement du trafic entre eux VPCs. Pour cela, plusieurs AWS services peuvent être utilisés pour compléter vos composants réseau existants.

- AWS Transit Gateway — Vous devez d'abord envisager ce service réseau. Ce service sert de hub central pour le routage de vos connexions entre Amazon VPCs et Comptes AWS les réseaux locaux. Pour plus d'informations, voir Qu'est-ce qu'une passerelle de transit ? dans le guide Amazon VPC Transit Gateways.
- Support Amazon VPC et VPN : vous pouvez utiliser ce service pour créer des connexions site-to-site VPN afin de connecter des réseaux locaux à votre VPC. Pour plus d'informations, voir Qu'est-ce que c'est AWS Site-to-Site VPN ? dans le guide de AWS Site-to-Site VPN l'utilisateur.
- Amazon VPC — Vous pouvez utiliser le peering Amazon VPC pour vous aider à en connecter plusieurs VPCs, soit dans le même compte, soit entre plusieurs comptes. Pour plus d'informations, consultez Qu'est-ce que l'appairage de VPC ? dans le Guide d'appairage de VPC Amazon.

- Partagé VPCs : vous pouvez utiliser un VPC et des sous-réseaux VPC sur plusieurs comptes. AWS Pour plus d'informations, consultez la section Travailler avec le partage VPCs dans le guide de l'utilisateur Amazon VPC.

AWS services de dépannage réseau Amazon ECS

Les services et fonctionnalités suivants peuvent vous aider à mieux comprendre les configurations de votre réseau et de vos services. Vous pouvez utiliser ces informations pour résoudre les problèmes de réseau et optimiser au mieux vos services.

CloudWatch Informations sur les conteneurs

CloudWatch Container Insights collecte, agrège et résume les métriques et les journaux de vos applications conteneurisées et de vos microservices. Les métriques incluent l'utilisation de ressources telles que le processeur, la mémoire, le disque et le réseau. Ils sont disponibles dans des tableaux de bord CloudWatch automatiques. Pour plus d'informations, consultez la section Configuration de Container Insights sur Amazon ECS dans le guide de CloudWatch l'utilisateur Amazon.

AWS X-Ray

AWS X-Ray est un service de suivi que vous pouvez utiliser pour collecter des informations sur les requêtes réseau effectuées par votre application. Vous pouvez utiliser le SDK pour instrumenter votre application et capturer les délais et les codes de réponse du trafic entre vos services, et entre vos services et les points de terminaison des services. AWS Pour plus d'informations, veuillez consulter la rubrique Présentation de AWS X-Ray dans le Guide du développeur AWS X-Ray .

Vous pouvez également explorer AWS X-Ray des graphiques illustrant la manière dont vos services sont en réseau les uns avec les autres. Vous pouvez également les utiliser pour explorer des statistiques agrégées sur les performances de chaque service-to-service lien. Enfin, vous pouvez approfondir une transaction spécifique pour voir comment les segments représentant des appels réseau sont associés à cette transaction en particulier.

Vous pouvez utiliser ces fonctionnalités pour identifier s'il existe un goulot d'étranglement réseau ou si un service spécifique de votre réseau ne fonctionne pas comme prévu.

Journaux de flux VPC

Vous pouvez utiliser les journaux de flux Amazon VPC pour analyser les performances du réseau et résoudre les problèmes de connectivité. Lorsque les journaux de flux VPC sont activés, vous pouvez capturer un journal de toutes les connexions de votre VPC. Il s'agit notamment des connexions aux interfaces réseau associées à Elastic Load Balancing, Amazon RDS, aux passerelles NAT et à d'autres AWS services clés que vous pourriez utiliser. Pour plus d'informations, consultez la rubrique Journaux de flux VPC dans le Guide de l'utilisateur Amazon VPC.

Conseils de réglage du réseau

Il existe quelques paramètres que vous pouvez affiner afin d'améliorer votre réseau.

aucune limite de fichier

Si vous vous attendez à ce que votre application génère un trafic élevé et gère de nombreuses connexions simultanées, vous devez tenir compte du quota système pour le nombre de fichiers autorisés. Lorsqu'un grand nombre de sockets réseau sont ouverts, chacun d'eux doit être représenté par un descripteur de fichier. Si votre quota de descripteurs de fichiers est trop faible, cela limitera le nombre de sockets réseau. Cela entraîne des échecs de connexion ou des erreurs. Vous pouvez mettre à jour le quota spécifique au conteneur pour le nombre de fichiers dans la définition de tâche Amazon ECS. Si vous utilisez Amazon EC2 (au lieu de AWS Fargate), vous devrez peut-être également ajuster ces quotas sur votre EC2 instance Amazon sous-jacente.

réseau sysctl

Les paramètres `sysctl` réseau constituent une autre catégorie de paramètres réglables. Vous devez vous référer aux paramètres spécifiques de la distribution Linux de votre choix. La plupart de ces paramètres ajustent la taille des tampons de lecture et d'écriture. Cela peut être utile dans certaines situations lors de l'exécution d' EC2 instances Amazon à grande échelle contenant de nombreux conteneurs.

Optimisez le temps de lancement des tâches Amazon ECS

Afin d'accélérer le lancement de vos tâches, tenez compte des recommandations suivantes.

- Cache les images du conteneur et les instances binpack

  Si vous utilisez le type de EC2 lancement, vous pouvez configurer le comportement d'extraction de l'agent de conteneur Amazon ECS pour `ECS_IMAGE_PULL_BEHAVIOR` : `prefer-cached`

L'image est extraite à distance s'il n'y a pas d'image mise en cache. Dans le cas contraire, l'image mise en cache sur l'instance est utilisée. Le nettoyage automatique des images est désactivé pour le conteneur afin de garantir que l'image mise en cache n'est pas supprimée. Cela réduit le temps d'extraction des images pour les lancements ultérieurs. L'effet de la mise en cache est encore plus important lorsque vous avez une densité de tâches élevée dans vos instances de conteneur, que vous pouvez configurer à l'aide de la stratégie de `binpack` placement. La mise en cache des images de conteneur est particulièrement avantageuse pour les charges de travail basées sur Windows qui comportent généralement des images de conteneur de grande taille (des dizaines GBs). Lorsque vous utilisez la stratégie de `binpack` placement, vous pouvez également envisager d'utiliser le trunking Elastic Network Interface (ENI) pour placer davantage de tâches en mode `awsvpc` réseau sur chaque instance de conteneur. Le trunking ENI augmente le nombre de tâches que vous pouvez exécuter en `awsvpc` mode. Par exemple, une instance c5.large qui peut prendre en charge l'exécution simultanée de seulement 2 tâches peut exécuter jusqu'à 10 tâches avec le trunking ENI.

- Choisissez un mode réseau optimal

  Bien que le mode `awsvpc` réseau soit idéal dans de nombreux cas, ce mode réseau peut intrinsèquement augmenter la latence de lancement des tâches, car pour chaque tâche en `awsvpc` mode, les flux de travail Amazon ECS doivent fournir et associer un ENI en invoquant Amazon, EC2 APIs ce qui ajoute un surcoût de plusieurs secondes au lancement de vos tâches. En revanche, l'un des principaux avantages du mode `awsvpc` réseau est que chaque tâche possède un groupe de sécurité qui autorise ou refuse le trafic. Cela signifie que vous disposez d'une plus grande flexibilité pour contrôler les communications entre les tâches et les services à un niveau plus détaillé. Si la vitesse de déploiement est votre priorité, vous pouvez envisager d'utiliser `bridge` le mode pour accélérer le lancement des tâches. Pour de plus amples informations, veuillez consulter the section called “AWSVPC mode réseau”.

- Suivez le cycle de vie de lancement de vos tâches pour identifier les opportunités d'optimisation

  Il est souvent difficile de connaître le temps nécessaire au démarrage de votre application. Le lancement de votre image de conteneur, l'exécution de scripts de démarrage et d'autres configurations lors du démarrage de l'application peuvent prendre un temps surprenant. Vous pouvez utiliser le point de terminaison des métadonnées des tâches pour publier des métriques permettant de suivre le temps de démarrage de l'application, depuis `StartedAt` la réponse des métadonnées du conteneur jusqu'à la `StartedAt` durée de votre tâche ou service. Grâce à ces données, vous pouvez comprendre comment votre application contribue au temps de lancement total et identifier les domaines dans lesquels vous pouvez réduire les frais inutiles spécifiques à l'application et optimiser les images de vos conteneurs. Pour de plus amples informations,

veuillez consulter Meilleures pratiques de mise à l'échelle automatique et de gestion des capacités d'Amazon ECS.

- Choisissez un type d'instance optimal (pour le type de EC2 lancement)

  Le choix du type d'instance approprié dépend de la réservation de ressources (par exemple, processeur, mémoire) que vous configurez pour votre tâche. Par conséquent, lors du dimensionnement de l'instance, vous pouvez calculer le nombre de tâches pouvant être placées sur une seule instance. Un exemple simple de tâche bien placée est l'hébergement de 4 tâches nécessitant 0,5 vCPU et 2 Go de réservations de mémoire dans une instance m5.large (prenant en charge 2 vCPU et 8 Go de mémoire). Les réservations de cette définition de tâche tirent pleinement parti des ressources de l'instance.

Exploiter Amazon ECS à grande échelle

Lorsque vous commencez à exploiter Amazon ECS à grande échelle, réfléchissez à la manière dont les quotas de service et les limitations d'API pour Amazon ECS et ceux Services AWS qui s'intègrent à Amazon ECS peuvent vous affecter.

Rubriques

- Quotas de service Amazon ECS et limites de limitation des API
- Gérer les problèmes de régulation d'Amazon ECS

Quotas de service Amazon ECS et limites de limitation des API

Amazon ECS s'intègre à plusieurs d'entre eux Services AWS, notamment Elastic Load Balancing et Amazon EC2. AWS Cloud Map Grâce à cette intégration étroite, Amazon ECS inclut plusieurs fonctionnalités telles que l'équilibrage de charge des services, la découverte des services, la mise en réseau des tâches et le dimensionnement automatique des clusters. Amazon ECS et l'autre solution Services AWS qu'il intègre à tous les systèmes maintiennent les quotas de service et les limites de débit des API afin de garantir des performances et une utilisation cohérentes. Ces quotas de service empêchent également le provisionnement accidentel de ressources supérieures à celles nécessaires et protègent contre les actions malveillantes susceptibles d'augmenter votre facture.

En vous familiarisant avec vos quotas de service et les limites de débit des AWS API, vous pouvez planifier le dimensionnement de vos charges de travail sans vous soucier d'une dégradation

inattendue des performances. Pour plus d'informations, consultez les rubriques Quotas de service Amazon ECS et Régulation des demandes pour l'API Amazon ECS.

Lorsque vous dimensionnez vos charges de travail sur Amazon ECS, tenez compte du quota de service suivant. Pour savoir comment demander une augmentation des quotas de service, consultez Gérer votre Amazon ECS et vos quotas AWS Fargate de service dans le AWS Management Console.

- AWS Fargate possède des quotas qui limitent le nombre de tâches exécutées simultanément dans chacune d'elles Région AWS. Il existe des quotas pour les tâches On-Demand et Fargate Spot sur Amazon ECS. Chaque quota de service inclut également tous les pods Amazon EKS que vous utilisez sur Fargate. Pour plus d'informations sur les quotas Fargate, AWS Fargate consultez les quotas de service dans le guide de l'utilisateur d'Amazon Elastic Container Service pour. AWS Fargate
- Pour les tâches exécutées sur des EC2 instances Amazon, le nombre maximum d' EC2 instances Amazon que vous pouvez enregistrer pour chaque cluster est de 5 000. Si vous utilisez le dimensionnement automatique du cluster Amazon ECS avec un fournisseur de capacité de groupe Auto Scaling, ou si vous gérez vous-même les EC2 instances Amazon de votre cluster, ce quota peut devenir un obstacle au déploiement. Si vous avez besoin de plus de capacité, vous pouvez créer d'autres clusters ou demander une augmentation du quota de service.
- Si vous utilisez le dimensionnement automatique du cluster Amazon ECS avec un fournisseur de capacité de groupe Auto Scaling, tenez compte du `Tasks in the PROVISIONING state per cluster` quota lors du dimensionnement de vos services. Ce quota est le nombre maximum de tâches dans l'`PROVISIONING`état pour chaque cluster pour lesquelles les fournisseurs de capacité peuvent augmenter la capacité. Lorsque vous lancez un grand nombre de tâches simultanément, vous pouvez facilement atteindre ce quota. Par exemple, si vous déployez simultanément des dizaines de services, chacun comportant des centaines de tâches. Dans ce cas, le fournisseur de capacité doit lancer de nouvelles instances de conteneur pour placer les tâches lorsque la capacité du cluster est insuffisante. Pendant que le fournisseur de capacité lance des EC2 instances Amazon supplémentaires, le planificateur de services Amazon ECS continuera probablement à lancer des tâches en parallèle. Toutefois, cette activité peut être limitée en raison d'une capacité de cluster insuffisante. Le planificateur de services Amazon ECS met en œuvre une stratégie de ralentissement et de limitation exponentielle pour réessayer de placer des tâches lorsque de nouvelles instances de conteneur sont lancées. Par conséquent, les délais de déploiement ou de mise à l'échelle peuvent être plus lents. Pour éviter cette situation, vous pouvez planifier vos déploiements de services de l'une des manières suivantes. Déployez un grand nombre de tâches

qui ne nécessitent pas d'augmenter la capacité du cluster ou conservez de la capacité de cluster inutilisée pour le lancement de nouvelles tâches.

En plus de prendre en compte le quota de service Amazon ECS lors du dimensionnement de vos charges de travail, tenez également compte du quota de service pour Services AWS les autres services intégrés à Amazon ECS. La section suivante décrit en détail les principales limites de débit pour chaque service et fournit des recommandations pour faire face aux éventuels problèmes de régulation.

Elastic Load Balancing

Vous pouvez configurer vos services Amazon ECS pour utiliser Elastic Load Balancing afin de répartir le trafic de manière uniforme entre les tâches. Pour plus d'informations sur le choix d'un équilibreur de charge, consultez Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS.

Quotas du service Elastic Load Balancing

Lorsque vous dimensionnez vos charges de travail, tenez compte des quotas du service Elastic Load Balancing suivants. La plupart des quotas du service Elastic Load Balancing sont ajustables, et vous pouvez demander une augmentation dans la console Service Quotas.

Application Load Balancer

Lorsque vous utilisez un Application Load Balancer, selon votre cas d'utilisation, vous devrez peut-être demander une augmentation de quota pour :

- Le `Targets per Application Load Balancer` quota, c'est-à-dire le nombre de cibles situées derrière votre Application Load Balancer.
- Le `Targets per Target Group per Region` quota, c'est-à-dire le nombre de cibles derrière vos groupes cibles.

Pour plus d'informations, consultez la section Quotas pour vos équilibreurs de charge d'application.

Network Load Balancer

Le nombre de cibles que vous pouvez enregistrer auprès d'un Network Load Balancer est soumis à des limites plus strictes. Lorsque vous utilisez un Network Load Balancer, vous souhaiterez souvent

activer le support entre zones, ce qui implique des limites de dimensionnement supplémentaires quant `Targets per Availability Zone Per Network Load Balancer` au nombre maximum de cibles par zone de disponibilité pour chaque Network Load Balancer. Pour plus d'informations, consultez la section Quotas pour vos équilibreurs de charge réseau.

Limitation de l'API Elastic Load Balancing

Lorsque vous configurez un service Amazon ECS pour utiliser un équilibreur de charge, les contrôles de santé du groupe cible doivent réussir avant que le service ne soit considéré comme sain. Pour effectuer ces contrôles de santé, Amazon ECS invoque les opérations de l'API Elastic Load Balancing en votre nom. Si vous avez configuré un grand nombre de services avec des équilibreurs de charge dans votre compte, les déploiements de services peuvent être plus lents en raison de la limitation potentielle liée aux opérations de l'API`RegisterTarget`,`DeregisterTarget`, et `DescribeTargetHealth` Elastic Load Balancing. En cas de régulation, des erreurs de régulation apparaissent dans les messages d'événements de votre service Amazon ECS.

Si vous êtes confronté à une limitation d' AWS Cloud Map API, vous pouvez nous contacter Support pour obtenir des conseils sur la manière d'augmenter vos limites de limitation AWS Cloud Map d'API. Pour plus d'informations sur la surveillance et le dépannage de telles erreurs de régulation, consultez la section Gestion des problèmes de régulation.

Interfaces réseau Elastic

Lorsque vos tâches utilisent le mode `awsvpc` réseau, Amazon ECS fournit une interface ELASTIC (ENI) unique pour chaque tâche. Lorsque vos services Amazon ECS utilisent un équilibreur de charge Elastic Load Balancing, ces interfaces réseau sont également enregistrées en tant que cibles pour le groupe cible approprié défini dans le service.

Quotas de service d'interface réseau élastiques

Lorsque vous exécutez des tâches utilisant le mode `awsvpc` réseau, une interface Elastic Network unique est associée à chaque tâche. Si ces tâches doivent être atteintes via Internet, attribuez une adresse IP publique à l'interface elastic network pour ces tâches. Lorsque vous dimensionnez vos charges de travail Amazon ECS, tenez compte de ces deux quotas importants :

- Le `Network interfaces per Region` quota qui est le nombre maximum d'interfaces réseau dans et Région AWS pour votre compte.
- Le `Elastic IP addresses per Region` quota qui est le nombre maximum d'adresses IP élastiques dans un Région AWS.

Ces deux quotas de service sont ajustables et vous pouvez demander une augmentation depuis votre console Service Quotas pour ces derniers. Pour plus d'informations, consultez la section Quotas de service Amazon VPC.

Pour les charges de travail Amazon ECS hébergées sur des EC2 instances Amazon, lors de l'exécution de tâches utilisant le mode `awsvpc` réseau, tenez compte du quota de `Maximum network interfaces` service, c'est-à-dire du nombre maximum d'instances réseau pour chaque EC2 instance Amazon. Ce quota limite le nombre de tâches que vous pouvez placer sur une instance. Vous ne pouvez pas ajuster le quota et celui-ci n'est pas disponible dans la console Service Quotas. Pour plus d'informations, consultez la section Adresses IP par interface réseau et par type d'instance dans le guide de EC2 l'utilisateur Amazon.

Bien que vous ne puissiez pas modifier le nombre d'interfaces réseau pouvant être associées à une EC2 instance Amazon, vous pouvez utiliser la fonctionnalité Elastic Network Interface trunking pour augmenter le nombre d'interfaces réseau disponibles. Par exemple, par défaut, une `c5.large` instance peut avoir jusqu'à trois interfaces réseau. L'interface réseau principale de l'instance est considérée comme une interface réseau. Vous pouvez donc associer deux interfaces réseau supplémentaires à l'instance. Comme chaque tâche utilisant le mode `awsvpc` réseau nécessite une interface réseau, vous ne pouvez généralement exécuter que deux tâches de ce type sur ce type d'instance. Cela peut entraîner une sous-utilisation de la capacité de votre cluster. Si vous activez Elastic Network Interface trunking, vous pouvez augmenter la densité de l'interface réseau afin de placer un plus grand nombre de tâches sur chaque instance. Lorsque le trunking est activé, une `c5.large` instance peut avoir jusqu'à 12 interfaces réseau. L'instance possède l'interface réseau principale et Amazon ECS crée et attache une interface réseau « tronc » à l'instance. Par conséquent, avec cette configuration, vous pouvez exécuter 10 tâches sur l'instance au lieu des deux tâches par défaut. Pour plus d'informations, consultez la section Elastic Network Interface trunking.

Limitation des API d'interface réseau élastique

Lorsque vous exécutez des tâches qui utilisent le mode `awsvpc` réseau, Amazon ECS s'appuie sur les Amazon suivants EC2 APIs. Chacun d'entre eux APIs a des limites d'API différentes. Pour plus d'informations, consultez la section Régulation des demandes pour l'API Amazon EC2 .

- CreateNetworkInterface
- AttachNetworkInterface
- DetachNetworkInterface
- DeleteNetworkInterface
- DescribeNetworkInterfaces

- DescribeVpcs
- DescribeSubnets
- DescribeSecurityGroups
- DescribeInstances

Si les appels d' EC2 API Amazon sont limités pendant les flux de travail de provisionnement de l'interface Elastic Network Interface, le planificateur de services Amazon ECS réessaie automatiquement avec des interruptions exponentielles. Ces tentatives automatiques peuvent parfois retarder le lancement des tâches, ce qui ralentit les vitesses de déploiement. En cas de limitation de l'API, le message apparaît `Operations are being throttled. Will try again later.` sur vos messages d'événement de service. Si vous respectez régulièrement les limites des EC2 API Amazon, vous pouvez nous contacter Support pour obtenir des conseils sur la manière d'augmenter vos limites de limitation des API. Pour plus d'informations sur la surveillance et le dépannage des erreurs de régulation, consultez la section Gestion des problèmes de régulation.

AWS Cloud Map

Amazon ECS Service Discovery permet AWS Cloud Map APIs de gérer les espaces de noms de vos services Amazon ECS. Si vos services comportent un grand nombre de tâches, tenez compte des recommandations suivantes. Pour plus d'informations, consultez les considérations relatives à la découverte des services Amazon ECS.

AWS Cloud Map quotas de service

Lorsque les services Amazon ECS sont configurés pour utiliser la découverte de services, le `Tasks per service` quota, qui est le nombre maximum de tâches pour le service, est affecté par le quota de AWS Cloud Map `Instances per service` service, qui est le nombre maximum d'instances pour ce service. En particulier, le quota de AWS Cloud Map service réduit le nombre de tâches que vous pouvez exécuter à un maximum de 1 000 tâches par service. Vous ne pouvez pas modifier le AWS Cloud Map quota. Pour plus d’informations, consultez Quotas de service AWS Cloud Map.

AWS Cloud Map Limitation de l'API

Amazon ECS appelle le`ListInstances`, `GetInstancesHealthStatusRegisterInstance`, et `DeregisterInstance` AWS Cloud Map APIs en votre nom. Ils facilitent la découverte des services et effectuent des bilans de santé lorsque vous lancez une tâche. Lorsque plusieurs services utilisant la découverte de services comportant un grand nombre de tâches sont déployés en même

temps, cela peut entraîner le dépassement des limites de limitation des AWS Cloud Map API. Dans ce cas, vous verrez probablement le message suivant : `Operations are being throttled. Will try again later` dans votre service Amazon ECS, des messages d'événement indiquant un ralentissement du déploiement et de la vitesse de lancement des tâches. AWS Cloud Map ne documente pas les limites de limitation pour ceux-ci. APIs Si vous êtes confronté à un ralentissement dû à ces derniers, vous pouvez nous contacter Support pour obtenir des conseils sur l'augmentation des limites de limitation de vos API. Pour plus de recommandations sur la surveillance et le dépannage de telles erreurs de régulation, consultez la section Gestion des problèmes de régulation.

Gérer les problèmes de régulation d'Amazon ECS

Les erreurs de régulation se répartissent en deux grandes catégories : la régulation synchrone et la régulation asynchrone.

Régulation synchrone

En cas de régulation synchrone, vous recevez immédiatement une réponse d'erreur de la part d'Amazon ECS. Cette catégorie apparaît généralement lorsque vous appelez Amazon ECS APIs lors de l'exécution de tâches ou de la création de services. Pour plus d'informations sur la régulation impliquée et les limites d'accélération pertinentes, consultez la section Régulation des demandes pour l'API Amazon ECS.

Lorsque votre application lance des demandes d'API, par exemple à l'aide du SDK AWS CLI ou d'un AWS SDK, vous pouvez remédier à la limitation des API. Pour ce faire, vous pouvez soit concevoir l'architecture de votre application de manière à gérer les erreurs, soit implémenter une stratégie de ralentissement exponentiel et d'instabilité avec une logique de nouvelle tentative pour les appels d'API. Pour plus d'informations, consultez les sections Expiration, nouvelles tentatives et interruption avec gigue.

Si vous utilisez un AWS SDK, la logique de nouvelle tentative automatique est intégrée et configurable.

Régulation asynchrone

La régulation asynchrone se produit en raison de flux de travail asynchrones dans lesquels Amazon ECS AWS CloudFormation ou Amazon pourrait APIs appeler en votre nom pour provisionner des ressources. Il est important de savoir lesquels Amazon ECS AWS APIs invoque en votre nom. Par exemple, l'`CreateNetworkInterface`API est invoquée pour les tâches qui utilisent le mode

`awsvpc` réseau, et l'`DescribeTargetHealth`API est invoquée lors de l'exécution de contrôles de santé pour les tâches enregistrées sur un équilibreur de charge.

Lorsque vos charges de travail atteignent une échelle considérable, ces opérations d'API peuvent être limitées. En d'autres termes, ils peuvent être suffisamment limités pour dépasser les limites imposées par Amazon ECS ou par celui Service AWS qui est appelé. Par exemple, si vous déployez des centaines de services, chacun comportant des centaines de tâches simultanément utilisant le mode `awsvpc` réseau, Amazon ECS invoque des opérations d' EC2 API Amazon telles que `CreateNetworkInterface` des opérations d'API Elastic Load Balancing telles que `RegisterTarget` ou `DescribeTargetHealth` pour enregistrer l'interface réseau élastique et l'équilibreur de charge, respectivement. Ces appels d'API peuvent dépasser les limites de l'API, ce qui entraîne des erreurs de régulation. Voici un exemple d'erreur de régulation d'Elastic Load Balancing incluse dans le message d'événement de service.

```
{
   "userIdentity":{
      "arn":"arn:aws:sts::111122223333:assumed-role/AWSServiceRoleForECS/ecs-service-
scheduler",
      "eventTime":"2022-03-21T08:11:24Z",
      "eventSource":"elasticloadbalancing.amazonaws.com",
      "eventName":" DescribeTargetHealth ",
      "awsRegion":"us-east-1",
      "sourceIPAddress":"ecs.amazonaws.com",
      "userAgent":"ecs.amazonaws.com",
      "errorCode":"ThrottlingException",
      "errorMessage":"Rate exceeded",
      "eventID":"0aeb38fc-229b-4912-8b0d-2e8315193e9c"
   }
}
```

Lorsque ces appels d'API partagent des limites avec le trafic d'autres API de votre compte, il peut être difficile de les surveiller, même s'ils sont émis sous forme d'événements de service.

Régulation du moniteur

Il est important d'identifier quelles demandes d'API sont limitées et qui les émet. Vous pouvez utiliser AWS CloudTrail lequel contrôle la régulation et s'intègre à CloudWatch Amazon Athena et Amazon. EventBridge Vous pouvez configurer CloudTrail pour envoyer des événements spécifiques à CloudWatch Logs. CloudWatch Logs : log insights, analyse et analyse les événements. Cela permet d'identifier les détails des événements de régulation, tels que l'utilisateur ou le rôle IAM qui a

effectué l'appel et le nombre d'appels d'API effectués. Pour plus d'informations, consultez la section Surveillance des fichiers CloudTrail journaux à l'aide de CloudWatch journaux.

Pour plus d'informations sur CloudWatch Logs Insights et des instructions sur la façon d'interroger les fichiers journaux, voir Analyse des données des CloudWatch journaux avec Logs Insights.

Amazon Athena vous permet de créer des requêtes et d'analyser des données à l'aide du langage SQL standard. Par exemple, vous pouvez créer une table Athena pour analyser CloudTrail les événements. Pour plus d'informations, voir Utilisation de la CloudTrail console pour créer une table Athena pour les CloudTrail journaux.

Après avoir créé une table Athena, vous pouvez utiliser des requêtes SQL telles que la suivante pour examiner `ThrottlingException` les erreurs.

Remplacez le *`user-input`* par vos valeurs.

```
select eventname, errorcode,eventsource,awsregion, useragent,COUNT(*) count
FROM cloudtrail_table-name
where errorcode = 'ThrottlingException'
AND eventtime between '2024-09-24T00:00:08Z' and '2024-09-23T23:15:08Z'
group by errorcode, awsregion, eventsource, useragent, eventname
order by count desc;
```

Amazon ECS envoie également des notifications d'événements à Amazon EventBridge. Il existe des événements de modification de l'état des ressources et des événements d'action de service. Ils incluent des événements de régulation des API tels `ECS_OPERATION_THROTTLED` que et. `SERVICE_DISCOVERY_OPERATION_THROTTLED` Pour de plus amples informations, veuillez consulter Événements liés aux actions du service Amazon ECS.

Ces événements peuvent être consommés par un service, par exemple AWS Lambda pour effectuer des actions en réponse. Pour de plus amples informations, veuillez consulter Gestion des événements Amazon ECS.

Si vous exécutez des tâches autonomes, certaines opérations d'API, par exemple, `RunTask` sont asynchrones et les opérations de nouvelle tentative ne sont pas effectuées automatiquement. Dans de tels cas, vous pouvez utiliser des services tels que EventBridge l'intégration pour réessayer des opérations limitées ou AWS Step Functions ayant échoué. Pour plus d'informations, consultez Gérer une tâche de conteneur (Amazon ECS, Amazon SNS).

CloudWatch À utiliser pour surveiller l'étranglement

CloudWatch propose une surveillance de l'utilisation de l'API sur l'espace de `Usage` noms sous By AWS Resource. Ces métriques sont enregistrées avec le type d'API et le nom de la métrique CallCount. Vous pouvez créer des alarmes qui démarreront chaque fois que ces mesures atteignent un certain seuil. Pour plus d'informations, consultez Visualisation de vos quotas de service et définition des alarmes.

CloudWatch propose également la détection des anomalies. Cette fonctionnalité utilise l'apprentissage automatique pour analyser et établir des bases de référence en fonction du comportement particulier de la métrique sur laquelle vous l'avez activée. En cas d'activité inhabituelle de l'API, vous pouvez utiliser cette fonctionnalité conjointement avec des CloudWatch alarmes. Pour plus d'informations, consultez la section Utilisation de la détection des CloudWatch anomalies.

En surveillant les erreurs de régulation de manière proactive, vous pouvez les contacter Support pour augmenter les limites de régulation pertinentes et également recevoir des conseils pour les besoins spécifiques de votre application.

Meilleures pratiques de mise à l'échelle automatique et de gestion des capacités d'Amazon ECS

Vous pouvez exécuter des charges de travail d'applications conteneurisées de toutes tailles sur Amazon ECS. Cela inclut des environnements de test minimaux et des environnements de production de grande envergure qui fonctionnent à l'échelle mondiale.

Avec Amazon ECS, comme pour tous les AWS services, vous ne payez que pour ce que vous utilisez. Lorsque vous concevez votre application de manière appropriée, vous pouvez réduire les coûts en consommant uniquement les ressources dont vous avez besoin au moment où vous en avez besoin.

Les recommandations suivantes vous montrent comment exécuter vos charges de travail Amazon ECS afin d'atteindre vos objectifs de niveau de service tout en fonctionnant de manière rentable.

Rubriques

- Détermination de la taille des tâches pour Amazon ECS
- Optimisation de la mise à l'échelle automatique du service Amazon ECS
- Capacité et disponibilité d'Amazon ECS

- Capacité du cluster Amazon ECS
- Choix de la taille des tâches Fargate pour Amazon ECS
- Accélérer le provisionnement des capacités des clusters Amazon ECS auprès des fournisseurs de capacité sur Amazon EC2

Détermination de la taille des tâches pour Amazon ECS

L'un des choix les plus importants lorsque vous déployez des conteneurs sur Amazon ECS concerne la taille de vos conteneurs et de vos tâches. La taille de vos conteneurs et de vos tâches est essentielle pour la mise à l'échelle et la planification des capacités.

Amazon ECS utilise deux mesures de ressources pour la capacité : le processeur et la mémoire. Amazon ECS mesure le processeur en unités de 1/1024 d'un vCPU complet (1024 unités équivalant à 1 vCPU entier). Amazon ECS mesure la mémoire en mégaoctets.

Dans la définition de votre tâche, vous pouvez déclarer des réservations et des limites de ressources.

Lorsque vous déclarez une réservation, vous déclarez la quantité minimale de ressources requise par une tâche. Votre tâche reçoit au moins la quantité de ressources que vous demandez. Votre application peut utiliser plus de processeur ou de mémoire que la réservation que vous déclarez. Toutefois, cela est soumis aux limites que vous avez également déclarées.

Le fait d'utiliser un montant supérieur au montant de la réservation est appelé « bursting ». L'éclatement signifie que votre application utilise plus de ressources que ce que vous avez réservé, mais reste dans les limites que vous avez déclarées. Amazon ECS garantit les réservations. Par exemple, si vous utilisez des EC2 instances Amazon pour fournir de la capacité, Amazon ECS ne place aucune tâche sur une instance dans laquelle il ne peut pas exécuter la réservation.

Une limite est la quantité maximale d'unités de processeur ou de mémoire que votre conteneur ou votre tâche peut utiliser. Si votre conteneur essaie d'utiliser plus de CPU que cette limite, Amazon ECS le limite. Si votre conteneur essaie d'utiliser plus de mémoire que cette limite, Amazon ECS l'arrête.

Le choix de ces valeurs peut s'avérer difficile. Les valeurs les mieux adaptées à votre application dépendent dans une large mesure des besoins en ressources de votre application.

Le test de charge de votre application est la clé d'une planification réussie des besoins en ressources. Les tests de charge vous aident à mieux comprendre les exigences de votre application.

Demandes apatrides

Pour les applications sans état qui évoluent horizontalement, telles qu'une application située derrière un équilibreur de charge, nous vous recommandons de déterminer d'abord la quantité de mémoire consommée par votre application lorsqu'elle traite des demandes.

Pour ce faire, vous pouvez utiliser des outils traditionnels tels que `ps` `outop`. Vous pouvez également utiliser des solutions de surveillance telles que CloudWatch Container Insights.

Lorsque vous déterminez une réserve de processeur, réfléchissez à la manière dont vous souhaitez adapter votre application aux besoins de votre entreprise.

Vous pouvez utiliser des réserves de processeur plus petites, telles que 256 unités de processeur (ou 1/4 de vCPU), pour effectuer une mise à l'échelle précise tout en minimisant les coûts. Mais ils risquent de ne pas évoluer assez rapidement pour répondre à des pics de demande importants.

Vous pouvez utiliser des réservations de processeur plus importantes pour augmenter et diminuer plus rapidement. Cela vous permet de faire face plus rapidement aux pics de demande. Cependant, les réservations de processeurs plus importantes coûtent plus cher.

Autres applications

Pour les applications qui ne sont pas évolutives horizontalement, telles que les travailleurs individuels ou les serveurs de base de données, la capacité disponible et le coût sont vos principaux facteurs à prendre en compte.

Choisissez la quantité de mémoire et de processeur en fonction des tests de charge qui indiquent que vous avez besoin pour gérer le trafic et atteindre votre objectif de niveau de service. Amazon ECS garantit que votre application est placée sur un hôte doté d'une capacité adéquate.

Optimisation de la mise à l'échelle automatique du service Amazon ECS

Un service Amazon ECS est un ensemble géré de tâches. Chaque service est associé à une définition de tâche, à un nombre de tâches souhaité et à une stratégie de placement facultative.

Le dimensionnement automatique du service Amazon ECS fonctionne via le service Application Auto Scaling. Application Auto Scaling utilise CloudWatch les métriques comme source pour le dimensionnement des métriques. Il utilise également des CloudWatch alarmes pour définir des seuils indiquant quand il faut étendre ou dédimensionner votre service.

Vous indiquez les seuils de mise à l'échelle. Vous pouvez définir une cible métrique, appelée échelle de suivi des cibles. Vous pouvez également spécifier des seuils, appelés échelonnement par étapes.

Une fois que vous avez configuré Application Auto Scaling, celui-ci calcule en permanence le nombre de tâches souhaité pour le service. Il indique également à Amazon ECS lorsque le nombre de tâches souhaité doit changer, soit en le redimensionnant, soit en le redimensionnant.

Pour utiliser efficacement le dimensionnement automatique des services, vous devez choisir une métrique de mise à l'échelle appropriée. Nous expliquons comment choisir une métrique dans les sections suivantes.

Caractérisation de votre application

Pour dimensionner correctement une application, vous devez connaître les conditions dans lesquelles vous devez effectuer une mise à l'échelle de votre application et à quel moment vous devez la redimensionner.

Essentiellement, vous devez étendre votre application si l'on prévoit que la demande sera supérieure à la capacité. À l'inverse, vous pouvez adapter votre application pour réduire les coûts lorsque les ressources dépassent la demande.

Identification d'une métrique d'utilisation

Pour effectuer une mise à l'échelle efficace, vous devez identifier une métrique indiquant l'utilisation ou la saturation. Cette métrique doit présenter les propriétés suivantes pour être utile à la mise à l'échelle.

- La métrique doit être corrélée à la demande. Lorsque vous maintenez les ressources stables mais que la demande change, la valeur de la métrique doit également changer. La métrique doit augmenter ou diminuer lorsque la demande augmente ou diminue.
- La valeur métrique doit évoluer proportionnellement à la capacité. Lorsque la demande reste constante, l'ajout de ressources supplémentaires doit entraîner une modification proportionnelle de la valeur de la métrique. Ainsi, le doublement du nombre de tâches devrait entraîner une diminution de 50 % de la métrique.

Le meilleur moyen d'identifier une métrique d'utilisation consiste à effectuer des tests de charge dans un environnement de pré-production tel qu'un environnement intermédiaire. Les solutions de test de charge commerciales et open source sont largement disponibles. Ces solutions peuvent généralement générer une charge synthétique ou simuler le trafic utilisateur réel.

Pour démarrer le processus de test de charge, vous devez d'abord créer des tableaux de bord pour les indicateurs d'utilisation de votre application. Ces indicateurs incluent l'utilisation du processeur, l'utilisation de la mémoire, I/O les opérations, la profondeur des I/O files d'attente et le débit du réseau. Vous pouvez collecter ces statistiques à l'aide d'un service tel que CloudWatch Container Insights. Vous pouvez également les collecter en utilisant Amazon Managed Service for Prometheus avec Amazon Managed Grafana. Au cours de ce processus, assurez-vous de collecter et de tracer des mesures relatives aux temps de réponse ou aux taux d'achèvement des travaux de votre application.

Lorsque vous chargez un test, commencez par un faible taux de demande ou d'insertion d'emplois. Maintenez ce taux constant pendant plusieurs minutes pour permettre à votre application de s'échauffer. Ensuite, augmentez lentement le taux et maintenez-le stable pendant quelques minutes. Répétez ce cycle en augmentant le taux à chaque fois jusqu'à ce que les délais de réponse ou de traitement de votre demande soient trop lents pour atteindre vos objectifs de niveau de service ()SLOs.

Pendant le test de chargement, examinez chacune des mesures d'utilisation. Les indicateurs qui augmentent en fonction de la charge sont les meilleurs candidats pour vous servir de meilleurs indicateurs d'utilisation.

Identifiez ensuite la ressource qui atteint la saturation. Dans le même temps, examinez également les indicateurs d'utilisation pour voir lequel s'aplatit en premier à un niveau élevé. Vous pouvez également examiner lequel atteint le pic, puis bloque votre application en premier. Par exemple, si l'utilisation du processeur passe de 0 % à 70 à 80 % à mesure que vous ajoutez de la charge, puis qu'elle reste à ce niveau une fois que vous ajoutez encore plus de charge, on peut affirmer sans risque de se tromper que le processeur est saturé. Selon l'architecture du processeur, il se peut qu'il n'atteigne jamais 100 %. Supposons, par exemple, que l'utilisation de la mémoire augmente à mesure que vous ajoutez de la charge, puis que votre application se bloque soudainement lorsqu'elle atteint la limite de mémoire de la tâche ou de l' EC2 instance Amazon. Dans ce cas, il est probable que la mémoire ait été entièrement consommée. Plusieurs ressources peuvent être consommées par votre application. Par conséquent, choisissez la métrique qui représente la ressource qui s'épuise en premier.

Enfin, réessayez de tester la charge après avoir doublé le nombre de tâches ou d' EC2instances Amazon. Supposons que l'indicateur clé augmente ou diminue de moitié par rapport au taux précédent. Si tel est le cas, la métrique est proportionnelle à la capacité. Il s'agit d'un bon indicateur d'utilisation pour le dimensionnement automatique.

Examinons maintenant ce scénario hypothétique. Supposons que vous testiez le chargement d'une application et que vous constatiez que l'utilisation du processeur atteint finalement 80 % à 100 demandes par seconde. Lorsque vous ajoutez de la charge, l'utilisation du processeur n'augmente plus. Cependant, cela ralentit la réponse de votre application. Ensuite, vous exécutez à nouveau le test de charge, en doublant le nombre de tâches tout en maintenant le taux à sa valeur maximale précédente. Si vous constatez que l'utilisation moyenne du processeur tombe à environ 40 %, l'utilisation moyenne du processeur est un bon candidat pour une métrique de mise à l'échelle. En revanche, si l'utilisation du processeur reste à 80 % après l'augmentation du nombre de tâches, l'utilisation moyenne du processeur n'est pas un bon indicateur de mise à l'échelle. Dans ce cas, vous devez effectuer des recherches supplémentaires pour trouver une métrique appropriée.

Modèles d'application et propriétés de mise à l'échelle courants

Vous pouvez exécuter des logiciels de toutes sortes sur AWS. De nombreuses charges de travail sont développées en interne, tandis que d'autres sont basées sur des logiciels open source populaires. Quelle que soit leur origine, nous avons observé certains modèles de conception courants pour les services. La manière dont vous effectuez une mise à l'échelle efficace dépend en grande partie du modèle.

Le serveur efficace lié au processeur

Le serveur efficace lié au processeur n'utilise pratiquement aucune ressource autre que le processeur et le débit du réseau. Chaque demande peut être traitée par l'application seule. Les demandes ne dépendent pas d'autres services tels que les bases de données. L'application peut traiter des centaines de milliers de demandes simultanées et peut en utiliser plusieurs efficacement CPUs pour ce faire. Chaque demande est traitée par un thread dédié avec une faible charge de mémoire ou une boucle d'événements asynchrone s'exécute sur chaque processeur qui traite les demandes. Chaque réplique de l'application est également capable de traiter une demande. La seule ressource susceptible d'être épuisée avant le processeur est la bande passante réseau. Dans les services liés au processeur, l'utilisation de la mémoire, même au débit maximal, ne représente qu'une fraction des ressources disponibles.

Vous pouvez utiliser la mise à l'échelle automatique basée sur le processeur pour ce type d'application. L'application bénéficie d'une flexibilité maximale en termes de mise à l'échelle. Vous pouvez le redimensionner verticalement en lui fournissant des EC2 instances Amazon ou Fargate v CPUs plus grandes. Vous pouvez également le redimensionner horizontalement en ajoutant d'autres répliques. L'ajout de répliques supplémentaires, ou le doublement de la taille de l'instance, réduit de moitié l'utilisation moyenne du processeur par rapport à la capacité.

Si vous utilisez les EC2 capacités d'Amazon pour cette application, pensez à la placer sur des instances optimisées pour le calcul, telles que la famille `c5` or`c6g`.

Le serveur économe en mémoire

Le serveur limité à la mémoire efficace alloue une quantité importante de mémoire par demande. En cas de simultanéité maximale, mais pas nécessairement de débit, la mémoire est épuisée avant que les ressources du processeur ne soient épuisées. La mémoire associée à une demande est libérée lorsque celle-ci prend fin. Des demandes supplémentaires peuvent être acceptées dans la limite de la mémoire disponible.

Vous pouvez utiliser le dimensionnement automatique basé sur la mémoire pour ce type d'application. L'application bénéficie d'une flexibilité maximale en termes de mise à l'échelle. Vous pouvez le dimensionner à la fois verticalement en lui fournissant des ressources de mémoire Amazon EC2 ou Fargate plus importantes. Vous pouvez également le redimensionner horizontalement en ajoutant d'autres répliques. L'ajout de répliques supplémentaires ou le doublement de la taille de l'instance peut réduire de moitié l'utilisation moyenne de la mémoire par rapport à la capacité.

Si vous utilisez la EC2 capacité d'Amazon pour cette application, pensez à la placer sur des instances optimisées pour la mémoire, telles que la famille `r5` or`r6g`.

Certaines applications limitées en mémoire ne libèrent pas la mémoire associée à une demande lorsqu'elle se termine, de sorte qu'une réduction de la simultanéité n'entraîne pas une réduction de la mémoire utilisée. Pour cela, nous vous déconseillons d'utiliser le dimensionnement basé sur la mémoire.

Le serveur basé sur les travailleurs

Le serveur basé sur les travailleurs traite une demande pour chaque thread de travail individuel l'une après l'autre. Les threads de travail peuvent être des threads légers, tels que des threads POSIX. Il peut également s'agir de threads plus lourds, tels que des processus UNIX. Quel que soit le thread, l'application peut toujours prendre en charge une simultanéité maximale. Généralement, la limite de simultanéité est définie proportionnellement aux ressources de mémoire disponibles. Si la limite de simultanéité est atteinte, l'application place des demandes supplémentaires dans une file d'attente. Si la file d'attente du backlog est débordée, l'application rejette immédiatement les demandes entrantes supplémentaires. Les applications courantes qui correspondent à ce modèle incluent le serveur Web Apache et Gunicorn.

La simultanéité des demandes est généralement le meilleur indicateur pour dimensionner cette application. Comme il existe une limite de simultanéité pour chaque réplique, il est important de procéder à une mise à l'échelle avant que la limite moyenne ne soit atteinte.

Le meilleur moyen d'obtenir des mesures de simultanéité des demandes est de demander à votre application de les communiquer à CloudWatch. Chaque réplique de votre application peut publier le nombre de demandes simultanées sous forme de métrique personnalisée à une fréquence élevée. Nous recommandons de régler la fréquence au moins une fois par minute. Une fois que plusieurs rapports ont été collectés, vous pouvez utiliser la simultanéité moyenne comme mesure de mise à l'échelle. Vous calculez cette métrique en divisant la simultanéité totale par le nombre de répliques. Par exemple, si la simultanéité totale est de 1 000 et que le nombre de répliques est de 10, la simultanéité moyenne est de 100.

Si votre application se trouve derrière un Application Load Balancer, vous pouvez également utiliser la `ActiveConnectionCount` métrique de l'équilibreur de charge comme facteur dans la métrique de dimensionnement. Vous devez diviser la `ActiveConnectionCount` métrique par le nombre de répliques pour obtenir une valeur moyenne. Vous devez utiliser la valeur moyenne pour la mise à l'échelle, par opposition à la valeur de comptage brute.

Pour que cette conception fonctionne au mieux, l'écart type de latence de réponse doit être faible à de faibles taux de requêtes. Nous recommandons que, pendant les périodes de faible demande, la plupart des demandes reçoivent une réponse dans un court laps de temps, et il n'y a pas beaucoup de demandes dont le délai de réponse est nettement supérieur à la moyenne. Le temps de réponse moyen doit être proche du 95e percentile. Dans le cas contraire, des dépassements de files d'attente pourraient en résulter. Cela entraîne des erreurs. Nous vous recommandons de fournir des répliques supplémentaires si nécessaire pour atténuer le risque de débordement.

Le serveur d'attente

Le serveur en attente effectue un certain traitement pour chaque demande, mais son fonctionnement dépend fortement d'un ou de plusieurs services en aval. Les applications de conteneurs font souvent un usage intensif des services en aval tels que les bases de données et autres services d'API. La réponse de ces services peut prendre un certain temps, en particulier dans les scénarios de haute capacité ou de concurrence élevée. En effet, ces applications ont tendance à utiliser peu de ressources du processeur et à utiliser leur simultanéité maximale en termes de mémoire disponible.

Le service d'attente convient soit au modèle de serveur limité à la mémoire, soit au modèle de serveur basé sur le travailleur, selon la conception de l'application. Si la simultanéité de l'application est limitée uniquement par la mémoire, l'utilisation moyenne de la mémoire doit être utilisée comme

mesure de mise à l'échelle. Si la simultanéité de l'application est basée sur une limite de travail, la simultanéité moyenne doit être utilisée comme mesure de mise à l'échelle.

Le serveur basé sur Java

Si votre serveur basé sur Java dépend du processeur et s'adapte proportionnellement aux ressources du processeur, il convient peut-être au modèle de serveur efficace lié au processeur. Si tel est le cas, l'utilisation moyenne du processeur peut être appropriée comme mesure de mise à l'échelle. Cependant, de nombreuses applications Java ne sont pas liées au processeur, ce qui les rend difficiles à adapter.

Pour de meilleures performances, nous vous recommandons d'allouer autant de mémoire que possible au segment de machine virtuelle Java (JVM). Les versions récentes de la JVM, y compris la mise à jour 191 ou ultérieure de Java 8, définissent automatiquement la taille du tas aussi grande que possible pour tenir dans le conteneur. Cela signifie qu'en Java, l'utilisation de la mémoire est rarement proportionnelle à l'utilisation des applications. À mesure que le taux de demandes et la simultanéité augmentent, l'utilisation de la mémoire reste constante. Pour cette raison, nous ne recommandons pas de dimensionner les serveurs Java en fonction de l'utilisation de la mémoire. Nous recommandons plutôt une mise à l'échelle en fonction de l'utilisation du processeur.

Dans certains cas, les serveurs basés sur Java sont épuisés avant d'épuiser le processeur. Si votre application risque de s'épuiser en cas de forte simultanéité, le nombre moyen de connexions constitue le meilleur indicateur de mise à l'échelle. Si votre application est susceptible de s'épuiser en tas à haut débit, le taux de requêtes moyen est le meilleur indicateur de mise à l'échelle.

Serveurs utilisant d'autres environnements d'exécution collectés avec des déchets

De nombreuses applications serveur sont basées sur des environnements d'exécution qui collectent les déchets, tels que .NET et Ruby. Ces applications serveur peuvent correspondre à l'un des modèles décrits précédemment. Cependant, comme pour Java, nous ne recommandons pas de dimensionner ces applications en fonction de la mémoire, car l'utilisation moyenne de la mémoire observée n'est souvent pas corrélée au débit ou à la simultanéité.

Pour ces applications, nous vous recommandons de dimensionner l'utilisation du processeur si l'application est liée au processeur. Dans le cas contraire, nous vous recommandons d'effectuer une mise à l'échelle en fonction du débit moyen ou de la simultanéité moyenne, en fonction des résultats de vos tests de charge.

Processeurs de tâches

De nombreuses charges de travail impliquent un traitement de tâches asynchrone. Il s'agit notamment des applications qui ne reçoivent pas de demandes en temps réel, mais qui s'abonnent à une file d'attente pour recevoir des offres d'emploi. Pour ces types d'applications, la mesure de dimensionnement appropriée est presque toujours la profondeur de la file d'attente. La croissance des files d'attente indique que le travail en attente dépasse la capacité de traitement, tandis qu'une file d'attente vide indique qu'il y a plus de capacité que de travail à effectuer.

AWS les services de messagerie, tels qu'Amazon SQS et Amazon Kinesis Data Streams, fournissent des CloudWatch métriques qui peuvent être utilisées pour le dimensionnement. Pour Amazon SQS, `ApproximateNumberOfMessagesVisible` c'est le meilleur indicateur. Pour Kinesis Data Streams, pensez à utiliser `MillisBehindLatest` la métrique publiée par la Kinesis Client Library (KCL). Cette métrique doit être moyennée pour tous les consommateurs avant de l'utiliser à des fins de mise à l'échelle.

Capacité et disponibilité d'Amazon ECS

La disponibilité des applications est essentielle pour garantir une expérience sans erreur et minimiser le temps de latence des applications. La disponibilité dépend du fait de disposer de ressources accessibles et d'une capacité suffisante pour répondre à la demande. AWS fournit plusieurs mécanismes pour gérer la disponibilité. Pour les applications que vous hébergez sur Amazon ECS, il s'agit notamment de l'autoscaling et des zones de disponibilité (AZs). L'autoscaling gère le nombre de tâches ou d'instances en fonction des métriques que vous définissez, tandis que les zones de disponibilité vous permettent d'héberger votre application dans des emplacements isolés mais géographiquement proches.

Comme pour la taille des tâches, la capacité et la disponibilité présentent certains compromis que vous devez prendre en compte. Idéalement, la capacité devrait être parfaitement adaptée à la demande. Il y aurait toujours juste assez de capacité pour traiter les demandes et traiter les tâches afin d'atteindre les objectifs de niveau de service (SLOs), notamment une faible latence et un faible taux d'erreur. La capacité ne serait jamais trop élevée, ce qui entraînerait des coûts excessifs ; elle ne serait pas non plus trop faible, ce qui entraînerait une latence et des taux d'erreur élevés.

La mise à l'échelle automatique est un processus latent. Tout d'abord, vous CloudWatch devez recevoir des métriques en temps réel. Ensuite, il CloudWatch faut les agréger pour les analyser, ce qui peut prendre plusieurs minutes en fonction de la granularité de la métrique. CloudWatch compare les indicateurs aux seuils d'alarme pour identifier une pénurie ou un excès de ressources. Pour éviter

toute instabilité, vous devez configurer les alarmes de manière à ce que le seuil défini soit dépassé pendant quelques minutes avant que l'alarme ne se déclenche. Il faut également du temps pour configurer de nouvelles tâches et mettre fin à des tâches qui ne sont plus nécessaires.

En raison de ces retards potentiels dans le système, vous devez conserver une certaine marge de manœuvre en surprovisionnant. Le surprovisionnement peut aider à faire face à des pics de demande à court terme. Cela permet également à votre application de répondre à des demandes supplémentaires sans atteindre la saturation. Comme bonne pratique, vous pouvez définir votre objectif de mise à l'échelle entre 60 et 80 % d'utilisation. Cela permet à votre application de mieux gérer les pics de demande alors que de la capacité supplémentaire est encore en cours de provisionnement.

Une autre raison pour laquelle nous vous recommandons de surprovisionner est de pouvoir réagir rapidement aux défaillances de la zone de disponibilité. AWS recommande que les charges de travail de production soient traitées à partir de plusieurs zones de disponibilité. En effet, en cas de défaillance d'une zone de disponibilité, les tâches exécutées dans les zones de disponibilité restantes peuvent toujours répondre à la demande. Si votre application s'exécute dans deux zones de disponibilité, vous devez doubler le nombre normal de tâches. Cela vous permet de fournir une capacité immédiate en cas de panne potentielle. Si votre application s'exécute dans trois zones de disponibilité, nous vous recommandons d'exécuter 1,5 fois le nombre normal de tâches. C'est-à-dire, exécutez trois tâches pour deux qui sont nécessaires au service ordinaire.

Maximiser la vitesse de mise à l'échelle

La mise à l'échelle automatique est un processus réactif dont l'effet prend du temps. Cependant, il existe des moyens de minimiser le temps nécessaire à la mise à l'échelle.

Minimisez la taille de l'image. Les images plus grandes sont plus longues à télécharger depuis un référentiel d'images et à décompresser. Par conséquent, réduire la taille des images réduit le temps nécessaire au démarrage d'un conteneur. Pour réduire la taille de l'image, vous pouvez suivre ces recommandations spécifiques :

- Si vous pouvez créer un binaire statique ou utiliser Golang, créez votre image `FROM` scratch et incluez uniquement votre application binaire dans l'image résultante.
- Utilisez des images de base minimisées provenant de fournisseurs de distribution en amont, tels qu'Amazon Linux ou Ubuntu.
- N'incluez aucun artefact de construction dans votre image finale. L'utilisation de versions en plusieurs étapes peut y contribuer.

- Des `RUN` scènes compactes dans la mesure du possible. Chaque `RUN` étape crée une nouvelle couche d'image, ce qui entraîne un aller-retour supplémentaire pour télécharger la couche. Une `RUN` étape comportant plusieurs commandes jointes `&&` comporte moins de couches qu'une étape comportant plusieurs `RUN` étapes.
- Si vous souhaitez inclure des données, telles que des données d'inférence ML, dans votre image finale, incluez uniquement les données nécessaires pour démarrer et commencer à desservir le trafic. Si vous récupérez des données à la demande depuis Amazon S3 ou un autre système de stockage sans affecter le service, stockez plutôt vos données dans ces emplacements.

Gardez vos images à portée de main. Plus la latence du réseau est élevée, plus le téléchargement de l'image est long. Hébergez vos images dans un référentiel situé dans la même AWS région que celle dans laquelle se trouve votre charge de travail. Amazon ECR est un référentiel d'images hautes performances disponible dans toutes les régions dans lesquelles Amazon ECS est disponible. Évitez d'utiliser Internet ou un lien VPN pour télécharger des images de conteneurs. L'hébergement de vos images dans la même région améliore la fiabilité globale. Il atténue le risque de problèmes de connectivité réseau et de disponibilité dans une autre région. Vous pouvez également implémenter la réplication entre régions Amazon ECR pour y parvenir.

Réduisez les seuils de vérification de l'état de l'équilibreur de charge. Les équilibreurs de charge vérifient l'état de santé avant d'envoyer du trafic à votre application. La configuration du contrôle de santé par défaut pour un groupe cible peut prendre 90 secondes ou plus. Pendant ce temps, l'équilibreur de charge vérifie l'état de santé et reçoit les demandes. La réduction de l'intervalle entre les vérifications de santé et du nombre de seuils peut permettre à votre application d'accepter le trafic plus rapidement et de réduire la charge sur les autres tâches.

Tenez compte des performances de démarrage à froid. Certaines applications utilisent des environnements d'exécution tels que la compilation Java Perform Just-In-Time (JIT). Le processus de compilation, au moins lorsqu'il démarre, peut montrer les performances de l'application. Une solution consiste à réécrire les parties critiques de votre charge de travail en termes de latence dans des langages qui n'affectent pas les performances de démarrage à froid.

Utilisez le dimensionnement par étapes, et non des politiques de dimensionnement pour le suivi des cibles. Vous disposez de plusieurs options d'Application Auto Scaling pour les tâches Amazon ECS. Le suivi des cibles est le mode le plus simple à utiliser. Il vous suffit de définir une valeur cible pour une métrique, telle que l'utilisation moyenne du processeur. Ensuite, l'autoscaler gère automatiquement le nombre de tâches nécessaires pour atteindre cette valeur. Grâce à la mise à l'échelle par étapes, vous pouvez réagir plus rapidement aux variations de la demande, car vous

définissez les seuils spécifiques pour vos métriques de mise à l'échelle et le nombre de tâches à ajouter ou à supprimer lorsque les seuils sont dépassés. Et surtout, vous pouvez réagir très rapidement aux variations de la demande en réduisant la durée pendant laquelle une alarme de seuil est franchie. Pour plus d'informations, consultez Scalabilité automatique de service dans le Guide du développeur Amazon Elastic Container Service.

Si vous utilisez des EC2 instances Amazon pour fournir de la capacité de cluster, tenez compte des recommandations suivantes :

Utilisez des EC2 instances Amazon plus grandes et des volumes Amazon EBS plus rapides. Vous pouvez améliorer les vitesses de téléchargement et de préparation des images en utilisant une EC2 instance Amazon plus grande et un volume Amazon EBS plus rapide. Au sein d'une famille d' EC2 instances Amazon donnée, le débit maximal du réseau et d'Amazon EBS augmente à mesure que la taille de l'instance augmente (par exemple, de `m5.xlarge` à`m5.2xlarge`). En outre, vous pouvez également personnaliser les volumes Amazon EBS pour augmenter leur débit et leurs IOPS. Par exemple, si vous utilisez des `gp2` volumes, utilisez des volumes plus importants qui offrent un débit de base supérieur. Si vous utilisez des `gp3` volumes, spécifiez le débit et les IOPS lorsque vous créez le volume.

Utilisez le mode réseau en pont pour les tâches exécutées sur les EC2 instances Amazon. Les tâches qui utilisent le mode `bridge` réseau sur Amazon EC2 démarrent plus rapidement que celles qui utilisent le mode `awsvpc` réseau. Lorsque le mode `awsvpc` réseau est utilisé, Amazon ECS attache une interface ELASTIC (ENI) à l'instance avant de lancer la tâche. Cela introduit une latence supplémentaire. Il existe cependant plusieurs compromis à faire lors de l'utilisation d'un réseau de ponts. Ces tâches ne disposent pas de leur propre groupe de sécurité, ce qui a des répercussions sur l'équilibrage de charge. Pour plus d'informations, consultez la section Groupes cibles des équilibreurs de charge dans le guide de l'utilisateur d'Elastic Load Balancing.

Gérer les chocs liés à la demande

Certaines applications subissent des chocs soudains et importants en termes de demande. Cela se produit pour diverses raisons : un événement d'actualité, une grosse vente, un événement médiatique ou tout autre événement qui devient viral et entraîne une augmentation rapide et significative du trafic en très peu de temps. Si cela n'est pas planifié, la demande peut rapidement dépasser les ressources disponibles.

La meilleure façon de gérer les chocs de demande est de les anticiper et de planifier en conséquence. La mise à l'échelle automatique pouvant prendre du temps, nous vous recommandons de faire évoluer votre application avant que le choc de la demande ne commence. Pour de meilleurs

résultats, nous recommandons d'élaborer un plan d'affaires impliquant une étroite collaboration entre les équipes utilisant un calendrier partagé. L'équipe chargée de planifier l'événement doit travailler en étroite collaboration avec l'équipe chargée de l'application à l'avance. Cela donne à cette équipe suffisamment de temps pour établir un plan de planification clair. Ils peuvent planifier la capacité pour augmenter la capacité avant l'événement et pour l'augmenter après l'événement. Pour plus d'informations, voir Mise à l'échelle planifiée dans le Guide de l'utilisateur Application Auto Scaling..

Si vous avez un plan de Support aux entreprises, assurez-vous également de travailler avec votre responsable de compte technique (TAM). Votre TAM peut vérifier vos quotas de service et s'assurer que tous les quotas nécessaires sont augmentés avant le début de l'événement. De cette façon, vous n'atteignez aucun quota de service par accident. Ils peuvent également vous aider en proposant des services de préchauffage tels que des équilibreurs de charge pour garantir le bon déroulement de votre événement.

Il est plus difficile de gérer les chocs imprévus liés à la demande. Les chocs imprévus, s'ils sont d'une amplitude suffisante, peuvent rapidement faire en sorte que la demande dépasse la capacité. Cela peut également dépasser la capacité de réaction de l'autoscaling. La meilleure façon de se préparer à des chocs imprévus est de surapprovisionner les ressources. Vous devez disposer de suffisamment de ressources pour répondre à la demande maximale de trafic prévue à tout moment.

Le maintien d'une capacité maximale en prévision de chocs imprévus liés à la demande peut s'avérer coûteux. Pour atténuer l'impact sur les coûts, trouvez un indicateur avancé ou un événement qui prédit l'imminence d'un choc de demande important. Si la métrique ou l'événement fournit un préavis significatif de manière fiable, lancez le processus de scale-out immédiatement lorsque l'événement se produit ou lorsque la métrique dépasse le seuil spécifique que vous avez défini.

Si votre application est sujette à des chocs de demande soudains et imprévus, envisagez d'ajouter un mode haute performance à votre application qui sacrifie les fonctionnalités non critiques tout en préservant les fonctionnalités cruciales pour le client. Supposons, par exemple, que votre application puisse passer de la génération de réponses personnalisées coûteuses à la diffusion d'une page de réponse statique. Dans ce scénario, vous pouvez augmenter le débit de manière significative sans devoir dimensionner l'application.

Enfin, vous pouvez envisager de dissocier les services monolithiques afin de mieux faire face aux chocs de demande. Si votre application est un service monolithique coûteux à exécuter et lent à adapter, vous pouvez peut-être extraire ou réécrire des éléments critiques en termes de performances et les exécuter en tant que services distincts. Ces nouveaux services peuvent ensuite être étendus indépendamment des composants moins critiques. Le fait de disposer de la flexibilité nécessaire pour étendre les fonctionnalités critiques en termes de performances séparément des

autres parties de votre application peut à la fois réduire le temps nécessaire pour augmenter la capacité et contribuer à réduire les coûts.

Capacité du cluster Amazon ECS

Vous pouvez fournir de la capacité à un cluster Amazon ECS de plusieurs manières. Par exemple, vous pouvez lancer des EC2 instances Amazon et les enregistrer auprès du cluster au démarrage à l'aide de l'agent de conteneur Amazon ECS. Cependant, cette méthode peut s'avérer difficile car vous devez gérer vous-même le dimensionnement. Par conséquent, nous vous recommandons d'utiliser les fournisseurs de capacité Amazon ECS. Les fournisseurs de capacités gèrent le dimensionnement des ressources pour vous. Il existe trois types de fournisseurs de capacité : Amazon EC2, Fargate et Fargate Spot. Pour plus d'informations sur les fournisseurs de capacité Fargate, consultez les clusters Amazon ECS pour les charges de travail de type Fargate et pour le type de lancement, consultez les clusters Amazon ECS pour EC2 les charges de travail de type lancement. EC2

Les fournisseurs de capacité Fargate et Fargate Spot gèrent le cycle de vie des tâches Fargate à votre place. Fargate fournit une capacité à la demande et Fargate Spot fournit une capacité Spot. Lorsque vous lancez une tâche, Amazon ECS met à votre disposition une ressource Fargate. Cette ressource Fargate est fournie avec les unités de mémoire et de processeur qui correspondent directement aux limites de niveau de tâche que vous avez déclarées dans votre définition de tâche. Chaque tâche reçoit sa propre ressource Fargate, ce qui crée une relation 1:1 entre la tâche et les ressources de calcul.

Les tâches exécutées sur Fargate Spot sont susceptibles d'être interrompues. Les interruptions surviennent après un avertissement de deux minutes. Elles se produisent pendant les périodes de forte demande. Fargate Spot convient parfaitement aux charges de travail tolérantes aux interruptions, telles que les tâches par lots, les environnements de développement ou de préparation. Ils conviennent également à tout autre scénario où une haute disponibilité et une faible latence ne sont pas requises.

Vous pouvez exécuter des tâches Fargate Spot parallèlement à des tâches Fargate à la demande. En les utilisant ensemble, vous bénéficiez d'une capacité d'approvisionnement « en rafale » à moindre coût.

Amazon ECS peut également gérer la capacité des EC2 instances Amazon pour vos tâches. Chaque fournisseur EC2 de capacité Amazon est associé à un groupe Amazon EC2 Auto Scaling que vous spécifiez. Lorsque vous utilisez le fournisseur de EC2 capacité Amazon, le dimensionnement

automatique du cluster maintient la taille du groupe Amazon EC2 Auto Scaling afin de garantir que toutes les tâches planifiées peuvent être placées.

Choix de la taille des tâches Fargate pour Amazon ECS

Pour les définitions de AWS Fargate tâches, vous devez spécifier le processeur et la mémoire au niveau de la tâche, et vous n'avez pas besoin de prendre en compte les frais généraux. Vous pouvez également spécifier l'UC et la mémoire au niveau du conteneur pour les tâches Fargate. Toutefois, cette opération n'est pas obligatoire. Les limites de ressources doivent être supérieures ou égales à toutes les réservations que vous avez déclarées. Dans la plupart des cas, vous pouvez les définir en fonction de la somme des réservations de chacun des conteneurs déclarés dans votre définition de tâche. Ensuite, arrondissez le nombre à la taille de Fargate la plus proche. Pour plus d'informations sur les tailles disponibles, consultez la section Processeur des tâches et mémoire.

Accélérer le provisionnement des capacités des clusters Amazon ECS auprès des fournisseurs de capacité sur Amazon EC2

Les clients qui exécutent Amazon ECS sur Amazon EC2 peuvent tirer parti d'Amazon ECS Cluster Auto Scaling (CAS) pour gérer le dimensionnement des groupes Amazon EC2 Auto Scaling (ASG). Avec CAS, vous pouvez configurer Amazon ECS pour adapter automatiquement votre ASG et vous concentrer uniquement sur l'exécution de vos tâches. Amazon ECS veillera à ce que l'ASG évolue en fonction des besoins, sans qu'aucune autre intervention ne soit requise. Les fournisseurs de capacité Amazon ECS sont utilisés pour gérer l'infrastructure de votre cluster en veillant à ce qu'il y ait suffisamment d'instances de conteneur pour répondre aux demandes de votre application. Pour en savoir plus sur le fonctionnement d'Amazon ECS CAS, consultez Deep Dive on Amazon ECS Cluster Auto Scaling.

Comme le CAS repose sur une intégration CloudWatch basée sur l'intégration avec ASG pour ajuster la capacité du cluster, il présente une latence inhérente associée à la publication des CloudWatch métriques, `CapacityProviderReservation` au temps nécessaire pour que la métrique franchisse les CloudWatch alarmes (haute et faible) et au temps nécessaire à une EC2 instance Amazon récemment lancée pour se réchauffer. Vous pouvez prendre les mesures suivantes pour rendre le CAS plus réactif et accélérer les déploiements :

Dimensionnement par étapes du fournisseur de capacités

Les fournisseurs de capacité Amazon ECS finiront par fournir grow/shrink les instances de conteneur répondant aux exigences de votre application. Le nombre minimum d'instances qu'Amazon ECS lancera est fixé à 1 par défaut. Cela peut prolonger la durée de vos déploiements, si plusieurs

instances sont nécessaires pour placer vos tâches en attente. Vous pouvez augmenter le `minimumScalingStepSize`en utilisant l'API Amazon ECS pour augmenter le nombre minimum d'instances qu'Amazon ECS fait évoluer ou non à la fois. Une `maximumScalingStepSize`valeur trop faible peut limiter le nombre d'instances de conteneur augmentées ou dédimensionnées à la fois, ce qui peut ralentir vos déploiements.

> **Note**
>
> Cette configuration n'est actuellement disponible qu'en utilisant le `CreateCapacityProvider`ou `UpdateCapacityProvider` APIs.

Période de préchauffage de l'instance

La période de préchauffage de l'instance est la période après laquelle une EC2 instance Amazon récemment lancée peut contribuer aux CloudWatch statistiques du groupe Auto Scaling. Une fois la période de préchauffage spécifiée expirée, l'instance est prise en compte dans les métriques agrégées de l'ASG, et CAS procède à sa prochaine itération de calculs pour estimer le nombre d'instances requises.

La valeur par défaut `instanceWarmupPeriod`est de 300 secondes, que vous pouvez configurer à une valeur inférieure en utilisant le `CreateCapacityProvider`ou `UpdateCapacityProvider` APIs pour une mise à l'échelle plus réactive.

Capacité de réserve

Si votre fournisseur de capacité ne dispose d'aucune instance de conteneur pour placer des tâches, il doit augmenter (augmenter) la capacité du cluster en lançant des EC2 instances Amazon à la volée et en attendant qu'elles démarrent avant de pouvoir y lancer des conteneurs. Cela peut réduire considérablement le taux de lancement des tâches. Deux options s'offrent à vous.

Dans ce cas, le fait de disposer d'une EC2 capacité Amazon disponible déjà lancée et prête à exécuter des tâches augmentera le taux de lancement effectif des tâches. Vous pouvez utiliser la `Target Capacity` configuration pour indiquer que vous souhaitez conserver de la capacité inutilisée dans vos clusters. Par exemple, en définissant `Target Capacity` 80 %, vous indiquez que votre cluster a besoin de 20 % de capacité de réserve à tout moment. Cette capacité inutilisée peut permettre le lancement immédiat de toutes les tâches autonomes, garantissant ainsi que les lancements de tâches ne sont pas ralentis. L'inconvénient de cette approche est l'augmentation potentielle des coûts liés au maintien de la capacité inutilisée des clusters.

Une autre approche que vous pouvez envisager consiste à augmenter la marge de manœuvre de votre service, et non celle du fournisseur de capacité. Cela signifie qu'au lieu de réduire la `Target Capacity` configuration pour libérer de la capacité inutilisée, vous pouvez augmenter le nombre de répliques dans votre service en modifiant la métrique de dimensionnement du suivi des cibles ou les seuils de dimensionnement par étapes du dimensionnement automatique du service. Notez que cette approche ne sera utile que pour les charges de travail complexes, mais n'aura aucun effet lorsque vous déployez de nouveaux services et que vous passez de 0 à N tâches pour la première fois. Pour plus d'informations sur les politiques de dimensionnement associées, consultez les sections Politiques de dimensionnement de Target Tracking ou Politiques de dimensionnement par étapes

Accédez aux fonctionnalités d'Amazon ECS avec les paramètres du compte

Vous pouvez accéder aux paramètres de compte Amazon ECS pour activer ou désactiver des fonctions spécifiques. Pour chaque Région AWS, vous pouvez accepter ou refuser chaque paramètre de compte au niveau du compte ou pour un utilisateur ou un rôle spécifique.

Vous pouvez choisir d'activer ou de désactiver des fonctions spécifiques si l'une des options suivantes vous concerne :

- Un utilisateur ou un rôle peut accepter ou refuser des paramètres de compte spécifiques pour son compte individuel.
- Un utilisateur ou un rôle peut définir le paramètre d'acceptation et de refus par défaut pour tous les utilisateurs du compte.
- L'utilisateur root ou un utilisateur doté de privilèges d'administrateur peut accepter ou refuser un rôle ou un utilisateur spécifique sur le compte. Si le paramètre de compte de l'utilisateur root est modifié, cela modifie également le paramètre par défaut de tous les utilisateurs et rôles pour lesquels aucun paramètre de compte individuel n'a été sélectionné.

> **Note**
>
> Les utilisateurs fédérés héritent du paramètre de compte de l'utilisateur root et ne peuvent pas avoir de paramètres de compte définis séparément pour leur compte.

Les paramètres de compte suivants sont disponibles. Vous devez vous inscrire et vous désinscrire séparément pour chaque paramètre de compte.

Nom de la ressource	En savoir plus
`containerInsights`	Container Insights

`serviceLongArnFormat`
`taskLongArnFormat`
`containerInstanceLongArnFormat`
	Amazon Resource Names (ARNs) et IDs

`tagResourceAuthorization`	Autorisation de balisage
`fargateFIPSMode`	AWS Fargate Conformité à la norme fédérale de traitement de l'information (FIPS-140)
`fargateTaskRetirementWaitPeriod`	AWS Fargate temps d'attente pour la retraite des tâches
`guardDutyActivate`	Surveillance du temps d'exécution (GuardDuty intégration Amazon)
`dualStackIPv6`	IPv6 VPC à double pile
`awsvpcTrunking`	Augmenter les interfaces réseau des instances de conteneurs Linux
`defaultLogDriverMode`	Mode pilote de journal par défaut

Amazon Resource Names (ARNs) et IDs

Lorsque des ressources Amazon ECS sont créées, chacune d'elles se voit affecter un Amazon Resource Name (ARN) et un identificateur de ressource (ID) uniques. Si vous utilisez un outil de ligne de commande ou l'API Amazon ECS pour travailler avec Amazon ECS, des ressources ARNs ou IDs sont requises pour certaines commandes. Par exemple, si vous utilisez la AWS CLI commande stop-task pour arrêter une tâche, vous devez spécifier l'ARN ou l'ID de la tâche dans la commande.

Amazon ECS a introduit un nouveau format pour Amazon Resource Names (ARNs) et une nouvelle ressource IDs pour les services, les tâches et les instances de conteneur Amazon ECS. L'état d'acceptation de chaque type de ressource détermine le format Amazon Resource Name (ARN) utilisé par la ressource. Vous devez accepter le nouveau format ARN afin d'utiliser des fonctions telles que le balisage des ressources pour ce type de ressource.

Vous pouvez accepter ou refuser le nouveau nom Amazon Resource Name (ARN) et les ID de ressource dépend de la région. Actuellement, il est activé par défaut pour tout nouveau compte.

Vous pouvez accepter ou refuser le nouveau format d'Amazon Resource Name (ARN) et d'ID de ressource à tout moment. Une fois que vous vous êtes inscrit, toutes les nouvelles ressources que vous créez utilisent le nouveau format.

> **Note**
>
> Un ID de ressource ne change pas une fois qu'il a été créé. Par conséquent, le fait d'accepter ou de refuser le nouveau format n'affecte pas votre ressource IDs existante.

Les sections suivantes décrivent les modifications des formats d'ARN et d'ID de ressource. Pour plus d'informations sur la transition vers les nouveaux formats, consultez la FAQ sur Amazon Elastic Container Service.

Format Amazon Resource Name (ARN)

Certaines ressources ont un nom convivial, comme par exemple un service nommé `production`. Dans d'autres cas, vous devez définir une ressource à l'aide du format Amazon Resource Name (ARN). Le nouveau format ARN des tâches, services et instances de conteneur Amazon ECS inclut le nom du cluster. Pour de plus amples informations sur le nouveau format ARN, consultez Modification des paramètres du compte Amazon ECS.

Le tableau suivant présente le format actuel et le nouveau format pour chaque type de ressource :

Type de ressource	ARN
Instance de conteneur	`arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:container-instance/` *`container-instance-id`* actuel

Type de ressource	ARN
	Nouveau : `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:container-instance/` *`cluster-name`* `/`*`container-instance-id`
Amazon ECS service	`arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:service/` *`service-name`* actuel Nouveau : `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:service/` *`cluster-name`* `/`*`service-name`*
Tâches Amazon ECS	`arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:task/`*`task-id`* actuel Nouveau : `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:task/`*`cluster-name`* `/`*`task-id`*

Longueur des ID de ressource

Un ID de ressource est constitué d'une combinaison unique de lettres et de chiffres. Les nouveaux formats d'ID de ressource incluent des formats plus courts IDs pour les tâches Amazon ECS et les instances de conteneur. Le format d'ID de ressource actuel comporte 36 caractères. Les nouveaux IDs sont dans un format de 32 caractères qui n'inclut aucun trait d'union. Pour de plus amples informations sur le nouveau format d'ID de ressource, consultez Modification des paramètres du compte Amazon ECS.

L'argument par défaut est `enabled`.

Seules les ressources lancées après l'acceptation recevront le nouveau format d'ARN et d'ID de ressource. Toutes les ressources existantes ne sont pas affectées. Pour que les services et les tâches Amazon ECS passent aux nouveaux formats d'ARN et d'ID de ressource, vous devez recréer les services et les tâches. Pour faire passer une instance de conteneur au nouveau format d'ARN et d'ID de ressource, il est nécessaire de la vider et de lancer et d'enregistrer une nouvelle instance de conteneur sur le cluster.

> **Note**
>
> Les tâches lancées par un service Amazon ECS ne peuvent recevoir le nouveau format d'ARN et d'ID de ressource que si le service a été créé à compter du 16 novembre 2018 et si l'utilisateur qui a créé le service a accepté le nouveau format pour les tâches.

Calendrier relatif au format ARN et de l'ID de ressource

Le calendrier prévoyant des créneaux d'activation/rejet du nouveau format d'Amazon Resource Name (ARN) et d'ID de ressource pour les ressources Amazon ECS a pris fin le 1er avril 2021. Le nouveau format est activé par défaut pour tout nouveau compte. Toutes les nouvelles ressources créées reçoivent le nouveau format et vous ne pouvez plus vous désinscrire.

Container Insights

Le 2 décembre 2024, AWS a publié Container Insights avec une observabilité améliorée pour Amazon ECS. Cette version prend en charge une observabilité améliorée pour les clusters Amazon ECS utilisant les types de lancement Amazon EC2 et Fargate. Une fois que vous avez configuré Container Insights avec une observabilité améliorée sur Amazon ECS, Container Insights collecte automatiquement des données télémétriques détaillées sur l'infrastructure, du niveau du cluster au niveau du conteneur dans votre environnement, et affiche vos données dans des tableaux de bord qui vous présentent diverses mesures et dimensions. Vous pouvez ensuite utiliser ces out-of-the-box tableaux de bord sur la console Container Insights pour mieux comprendre l'état et les performances de vos conteneurs, et pour atténuer les problèmes plus rapidement en identifiant les anomalies.

Nous vous recommandons d'utiliser Container Insights avec une observabilité améliorée plutôt que Container Insights, car cela fournit une visibilité détaillée de votre environnement de conteneurs, réduisant ainsi le délai moyen de résolution. Pour plus d'informations, consultez Amazon ECS Container Insights avec des métriques d'observabilité améliorées dans le guide de l'Amazon CloudWatch utilisateur.

Le paramètre du `containerInsights` compte par défaut est`disabled`.

Container Insights avec une meilleure observabilité

Utilisez la commande suivante pour activer Container Insights avec une meilleure observabilité.

Réglez le paramètre du `containerInsights` compte sur`enhanced`.

```
aws ecs put-account-setting --name containerInsights --value enhanced
```

Exemple de sortie

```
{
    "setting": {
        "name": "containerInsights",
        "value": "enhanced",
        "principalArn": "arn:aws:iam::123456789012:johndoe",
         "type": user
    }
}
```

Après avoir défini ce paramètre de compte, tous les nouveaux clusters utilisent automatiquement Container Insights avec une meilleure observabilité. Utilisez la `update-cluster-settings` commande pour ajouter Container Insights avec une observabilité améliorée à un cluster existant, ou pour mettre à niveau un cluster de Container Insights à Container Insights avec une observabilité améliorée.

```
aws ecs update-cluster-settings --cluster cluster-name --settings
 name=containerInsights,value=enhanced
```

Vous pouvez également utiliser la console pour configurer Container Insights avec une meilleure observabilité. Pour de plus amples informations, veuillez consulter Modification des paramètres du compte Amazon ECS.

Container Insights

Lorsque vous définissez le paramètre du `containerInsights` compte sur`enabled`, Container Insights est activé par défaut pour tous les nouveaux clusters. Vous pouvez modifier les clusters existants en utilisant`update-cluster-settings`.

Pour utiliser Container Insights, définissez le paramètre du `containerInsights` compte sur`enabled`. Utilisez la commande suivante pour activer Container Insights.

```
aws ecs put-account-setting --name containerInsights --value enabled
```

Exemple de sortie

```
{
    "setting": {
        "name": "containerInsights",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:johndoe",
         "type": user
    }
}
```

Lorsque vous définissez le paramètre du `containerInsights` compte sur`enabled`, Container Insights est activé par défaut pour tous les nouveaux clusters. Utilisez la `update-cluster-settings` commande pour ajouter Container Insights à un cluster existant.

```
aws ecs update-cluster-settings --cluster cluster-name --settings
 name=containerInsights,value=enabled
```

Vous pouvez également utiliser la console pour configurer Container Insights. Pour de plus amples informations, veuillez consulter Modification des paramètres du compte Amazon ECS.

AWS Fargate Conformité à la norme fédérale de traitement de l'information (FIPS-140)

Fargate prend en charge la norme Federal Information Processing Standard (FIPS-140), qui spécifie les exigences de sécurité pour les modules cryptographiques qui protègent des informations sensibles. Il s'agit de la norme gouvernementale en vigueur aux États-Unis et au Canada, applicable aux systèmes qui doivent être conformes à la loi sur la gestion de la sécurité des informations fédérales (FISMA) ou au programme fédéral de gestion des risques et des autorisations (FedRAMP).

Le nom de la ressource est`fargateFIPSMode`.

L'argument par défaut est `disabled`.

Vous devez activer la conformité à la norme Federal Information Processing Standard (FIPS-140) sur Fargate. Pour de plus amples informations, veuillez consulter the section called “AWS Fargate Conformité à la norme FIPS-140”.

> **Important**
>
> Le paramètre de compte `fargateFIPSMode` ne peut être modifié qu'à l'aide de l'API Amazon ECS ou de la AWS CLI. Pour de plus amples informations, veuillez consulter Modification des paramètres du compte Amazon ECS.

Exécutez `put-account-setting-default` avec l'option `fargateFIPSMode` définie à `enabled`. Pour plus d'informations, put-account-setting-defaultconsultez le manuel Amazon Elastic Container Service API Reference.

- Vous pouvez utiliser la commande suivante pour activer la conformité à la norme FIPS-140.

```
aws ecs put-account-setting-default --name fargateFIPSMode --value enabled
```

Exemple de sortie

```
{
    "setting": {
        "name": "fargateFIPSMode",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:root",
         "type": user
    }
}
```

Vous pouvez exécuter `list-account-settings` pour afficher l'état actuel de conformité à la norme FIPS-140. Utilisez l'option `effective-settings` pour afficher les paramètres au niveau du compte.

```
aws ecs list-account-settings --effective-settings
```

Autorisation de balisage

Amazon ECS introduit l'autorisation de balisage pour la création de ressources. Les utilisateurs doivent disposer d'autorisations de balisage pour les actions qui créent la ressource, telles que`ecsCreateCluster`. Lorsque vous créez une ressource et que vous spécifiez des balises pour cette ressource, AWS effectue une autorisation supplémentaire pour vérifier qu'il existe

des autorisations pour créer des balises. Par conséquent, vous devez octroyer des autorisations explicites d'utiliser l'action `ecs:TagResource`. Pour de plus amples informations, veuillez consulter the section called “Baliser des ressources pendant la création”.

Pour activer l'autorisation de balisage, exécutez `put-account-setting-default` avec l'option `tagResourceAuthorization` définie sur `enable`. Pour plus d'informations, put-account-setting-defaultconsultez le manuel Amazon Elastic Container Service API Reference. Vous pouvez exécuter `list-account-settings` pour afficher l'état actuel de l'autorisation de balisage.

- Vous pouvez utiliser la commande suivante pour activer l'autorisation de balisage.

```
aws ecs put-account-setting-default --name tagResourceAuthorization --value on --
region region
```

Exemple de sortie

```
{
    "setting": {
        "name": "tagResourceAuthorization",
        "value": "on",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type": user
    }
}
```

Après avoir activé l'autorisation de balisage, vous devez configurer les autorisations appropriées pour permettre aux utilisateurs de baliser les ressources lors de leur création. Pour de plus amples informations, veuillez consulter the section called “Baliser des ressources pendant la création”.

Vous pouvez exécuter `list-account-settings` pour afficher l'état actuel de l'autorisation de balisage. Utilisez l'option `effective-settings` pour afficher les paramètres au niveau du compte.

```
aws ecs list-account-settings --effective-settings
```

Chronologie des autorisations de balisage

Vous pouvez vérifier si l'autorisation de balisage est active en exécutant `list-account-settings` pour afficher la valeur `tagResourceAuthorization`. Lorsque la valeur est égale à on, cela signifie

que l'autorisation de balisage est active. Pour plus d'informations, list-account-settingsconsultez le manuel Amazon Elastic Container Service API Reference.

Voici les dates importantes concernant l'autorisation de balisage.

- 18 avril 2023 : introduction de l'autorisation de balisage. Tous les comptes nouveaux et existants doivent adhérer pour utiliser la fonctionnalité. Vous pouvez choisir de commencer à utiliser l'autorisation de marquage. En adhérant, vous devez octroyer les autorisations appropriées.
- 9 février 2024 - 6 mars 2024 — L'autorisation de marquage est activée par défaut pour tous les nouveaux comptes et les comptes existants non concernés. Vous pouvez activer ou désactiver les paramètres du `tagResourceAuthorization` compte pour vérifier votre politique IAM.

  AWS a notifié les comptes concernés.

  Pour désactiver la fonctionnalité, `put-account-setting-default` exécutez-la avec l'`tagResourceAuthorization`option définie su`off`.
- 7 mars 2024 — Si vous avez activé l'autorisation de marquage, vous ne pouvez plus désactiver les paramètres du compte.

  Nous vous recommandons de terminer vos tests de politique IAM avant cette date.
- 29 mars 2024 — Tous les comptes utilisent l'autorisation de marquage. Le paramètre au niveau du compte ne sera plus disponible dans la console Amazon ECS ou. AWS CLI

AWS Fargate temps d'attente pour la retraite des tâches

AWS envoie des notifications lorsque des tâches Fargate s'exécutent sur une version de la plateforme dont la révision est marquée pour être supprimée. Pour de plus amples informations, veuillez consulter Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS .

AWS est responsable de l'application des correctifs et de la maintenance de l'infrastructure sous-jacente de AWS Fargate. Lorsqu'il est AWS déterminé qu'une mise à jour de sécurité ou d'infrastructure est nécessaire pour une tâche Amazon ECS hébergée sur Fargate, les tâches doivent être arrêtées et de nouvelles tâches lancées pour les remplacer. Vous pouvez configurer la période d'attente avant que les tâches ne soient retirées pour être corrigées. Vous avez la possibilité de mettre fin à la tâche immédiatement, d'attendre 7 jours calendaires ou d'attendre 14 jours calendaires.

Ce paramètre se trouve au niveau du compte.

Vous pouvez configurer l'heure à laquelle Fargate commence le retrait des tâches. Pour les charges de travail qui nécessitent l'application immédiate des mises à jour, choisissez le paramètre immédiat (`0`). Lorsque vous avez besoin de davantage de contrôle, par exemple lorsqu'une tâche ne peut être arrêtée que pendant une certaine période, configurez l'option 7 jours (`7`) ou 14 jours (`14`).

Nous vous recommandons de choisir une période d'attente plus courte afin de pouvoir accéder plus rapidement aux nouvelles versions de plateforme.

Configurez la période d'attente en exécutant `put-account-setting-default` ou `put-account-setting` en tant qu'utilisateur root ou administrateur. Utilisez l'option `fargateTaskRetirementWaitPeriod` pour le `name` et l'option `value` définie sur l'une des valeurs suivantes :

- `0`- AWS envoie la notification et commence immédiatement à supprimer les tâches concernées.
- `7`- AWS envoie la notification et attend 7 jours calendaires avant de commencer à supprimer les tâches concernées.
- `14` : AWS envoie la notification et attend 14 jours calendaires avant de commencer à retirer les tâches concernées.

La durée par défaut est de 7 jours.

Pour plus d'informations, consultez put-account-setting-defaultet consultez le put-account-settingmanuel Amazon Elastic Container Service API Reference.

Vous pouvez exécuter la commande suivante pour définir le délai d'attente à 14 jours.

```
aws ecs put-account-setting-default --name fargateTaskRetirementWaitPeriod --value 14
```

Exemple de sortie

```
{
    "setting": {
        "name": "fargateTaskRetirementWaitPeriod",
        "value": "14",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type: user"
    }
}
```

Vous pouvez exécuter `list-account-settings` pour afficher le temps d'attente actuel pour retirer la tâche Fargate. Utilisez l'option `effective-settings`.

```
aws ecs list-account-settings --effective-settings
```

Augmenter les interfaces réseau des instances de conteneurs Linux

Chaque tâche Amazon ECS qui utilise le mode `awsvpc` réseau reçoit sa propre interface Elastic Network (ENI), qui est attachée à l'instance de conteneur qui l'héberge. Le nombre d'interfaces réseau pouvant être associées à une EC2 instance Amazon est limité par défaut, et l'interface réseau principale compte pour une. Par exemple, par défaut, jusqu'à trois `c5.large` instances peuvent être ENIs associées. L'interface réseau principale de l'instance compte pour une seule, vous pouvez donc en attacher deux autres ENIs à l'instance. Comme chaque tâche utilisant le mode `awsvpc` réseau nécessite unENI, vous ne pouvez généralement exécuter que deux tâches de ce type sur ce type d'instance.

Amazon ECS prend en charge le lancement d'instances de conteneur avec une ENI densité accrue à l'aide des types d' EC2 instances Amazon pris en charge. Lorsque vous utilisez ces types d'instances et que vous activez le paramètre du `awsvpcTrunking` compte, d'autres ENIs sont disponibles sur les instances de conteneur récemment lancées. Cette configuration vous permet de placer plus de tâches sur chaque instance de conteneur.

Par exemple, une `c5.large` instance avec `awsvpcTrunking` une ENI limite accrue de douze. L'instance de conteneur a alors une interface réseau principale. Amazon ECS crée et attache une interface réseau « tronc » à l'instance de conteneur. Par conséquent, cette configuration vous permet de lancer dix tâches sur l'instance de conteneur au lieu des deux tâches actuellement possibles.

Surveillance du temps d'exécution (GuardDuty intégration Amazon)

Runtime Monitoring est un service intelligent de détection des menaces qui protège les charges de travail exécutées sur Fargate EC2 et les instances de conteneur en AWS surveillant en permanence l'activité des journaux et du réseau afin d'identifier les comportements malveillants ou non autorisés.

Le `guardDutyActivate` paramètre est en lecture seule dans Amazon ECS et indique si la surveillance du temps d'exécution est activée ou désactivée par votre administrateur de sécurité sur votre compte Amazon ECS. GuardDuty contrôle les paramètres de ce compte en votre nom. Pour plus d'informations, consultez Protéger les charges de travail Amazon ECS grâce à la surveillance du temps d'exécution.

Vous pouvez exécuter `list-account-settings` pour afficher le paramètre GuardDuty d'intégration actuel.

```
aws ecs list-account-settings
```

Exemple de sortie

```
{
    "setting": {
        "name": "guardDutyActivate",
        "value": "on",
        "principalArn": "arn:aws:iam::123456789012:doej",
        "type": aws-managed"
    }
}
```

IPv6 VPC à double pile

Amazon ECS permet de fournir aux tâches une IPv6 adresse en plus de l' IPv4 adresse privée principale.

Pour que les tâches reçoivent une IPv6 adresse, elles doivent utiliser le mode `awsvpc` réseau, être lancées dans un VPC configuré pour le mode double pile et le paramètre du `dualStackIPv6` compte doit être activé. Pour plus d'informations sur les autres exigences, reportez-vous aux sections Utilisation d'un VPC en mode double pile pour le type de EC2 lancement et Utilisation d'un VPC en mode double pile pour le type de lancement Fargate.

> **Important**
>
> Le paramètre de compte `dualStackIPv6` ne peut être modifié qu'à l'aide de l'API Amazon ECS ou de la AWS CLI. Pour de plus amples informations, veuillez consulter Modification des paramètres du compte Amazon ECS.

Si vous avez exécuté une tâche en mode `awsvpc` réseau dans un sous-réseau IPv6 activé entre le 1er octobre 2020 et le 2 novembre 2020, le paramètre de `dualStackIPv6` compte par défaut dans la région dans laquelle la tâche s'exécutait est`disabled`. Si cette condition n'est pas remplie, le paramètre de compte par défaut `dualStackIPv6` dans la région est `enabled`.

L'argument par défaut est `disabled`.

Mode pilote de journal par défaut

Amazon ECS prend en charge la définition d'un mode de livraison par défaut des messages de journal depuis un conteneur vers le pilote de journal choisi. Le mode de livraison affecte la stabilité de l'application lorsque le flux de journaux entre le conteneur et le pilote de journal est interrompu.

Le `defaultLogDriverMode` paramètre prend en charge deux valeurs : `blocking` et`non-blocking`. Pour plus d'informations sur ces modes de livraison, consultez le LogConfigurationmanuel Amazon Elastic Container Service API Reference.

Si vous ne spécifiez aucun mode de livraison dans la définition de votre conteneur`logConfiguration`, le mode que vous spécifiez à l'aide de ce paramètre de compte sera utilisé par défaut.

Le mode de livraison par défaut est`non-blocking`.

> **Note**
>
> Le 25 juin 2025, Amazon ECS a modifié le mode pilote de journal par défaut de à pour donner la priorité `blocking` `non-blocking` à la disponibilité des tâches plutôt qu'à la journalisation. Pour continuer à utiliser le `blocking` mode après cette modification, effectuez l'une des opérations suivantes :
>
> - Définissez l'`mode`option dans votre définition de conteneur `logConfiguration` comme`blocking`.
> - Réglez le paramètre du `defaultLogDriverMode` compte sur`blocking`.

Pour définir un mode pilote de journal par défaut sur`blocking`, vous pouvez exécuter la commande suivante.

```
aws ecs put-account-setting-default --name defaultLogDriverMode --value "blocking"
```

Afficher les paramètres du compte Amazon ECS à l'aide de la console

Consultez les paramètres de votre compte dans la console pour voir à quelles fonctionnalités vous avez accès.

> **Important**
>
> Les paramètres de compte `dualStackIPv6`, `fargateFIPSMode` et `fargateTaskRetirementWaitPeriod` ne peuvent être affichés ou modifiés qu'à l'aide de l' AWS CLI.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation en haut de l'écran, sélectionnez la région pour laquelle vous souhaitez afficher vos paramètres de compte.
3. Dans la page de navigation, choisissez Account Settings (Paramètres du compte).

Modification des paramètres du compte Amazon ECS

Modifiez les paramètres de votre compte pour accéder aux fonctionnalités d'Amazon ECS.

Le `guardDutyActivate` paramètre est en lecture seule dans Amazon ECS et indique si la surveillance du temps d'exécution est activée ou désactivée par votre administrateur de sécurité sur votre compte Amazon ECS. GuardDuty contrôle les paramètres de ce compte en votre nom. Pour plus d'informations, consultez Protéger les charges de travail Amazon ECS grâce à la surveillance du temps d'exécution.

> **Important**
>
> Les paramètres de compte `dualStackIPv6`, `fargateFIPSMode` et `fargateTaskRetirementWaitPeriod` ne peuvent être affichés ou modifiés qu'à l'aide de l' AWS CLI.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation en haut de l'écran, sélectionnez la région pour laquelle vous souhaitez afficher vos paramètres de compte.
3. Dans la page de navigation, choisissez Account Settings (Paramètres du compte).
4. Choisissez Mettre à jour.

5. (Facultatif) Pour augmenter ou diminuer le nombre de tâches que vous pouvez exécuter en mode réseau awsvpc pour chaque EC2 instance, sous AWSVPCTrunking, sélectionnez Trunking. AWSVPC
6. (Facultatif) Pour utiliser ou arrêter d'utiliser CloudWatch Container Insights par défaut pour les clusters, sous Observabilité de CloudWatch Container Insights, choisissez l'une des options suivantes :
   - Pour utiliser Container Insights avec une observabilité améliorée, choisissez Container Insights avec une observabilité améliorée.
   - Pour utiliser Container Insights, choisissez Container Insights.
   - Pour arrêter d'utiliser Container Insights, choisissez Désactivé.
7. (Facultatif) Pour activer ou désactiver l'autorisation de balisage, sous Autorisation de balisage des ressources, sélectionnez ou désactivez l'autorisation de balisage des ressources.
8. (Facultatif) Pour configurer un mode pilote de journal par défaut lorsqu'aucun mode de livraison de journal n'est défini dans un conteneur`logConfiguration`, sous Mode pilote de journal par défaut, choisissez l'une des options suivantes :
   - Pour définir le mode pilote de journal par défaut comme étant`blocking`, choisissez Blocage.
   - Pour définir le mode pilote de journal par défaut comme étant`non-blocking`, choisissez Non bloquant.
9. Sélectionnez Enregistrer les modifications.
10. Dans l'écran de confirmation, choisissez Confirm (Confirmer) pour enregistrer la sélection.

Étapes suivantes

Si vous avez activé Container Insights avec observabilité améliorée ou Container Insights, vous pouvez éventuellement mettre à jour vos clusters existants pour utiliser cette fonctionnalité. Pour de plus amples informations, veuillez consulter Mise à jour d'un cluster Amazon ECS.

Restauration des paramètres par défaut du compte Amazon ECS

Vous pouvez utiliser le AWS Management Console pour rétablir les paramètres par défaut de votre compte Amazon ECS.

L'option Revert to account default (Restaurer la valeur par défaut du compte) n'est disponible que lorsque les paramètres de votre compte ne sont plus les paramètres par défaut.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation en haut de l'écran, sélectionnez la région pour laquelle vous souhaitez afficher vos paramètres de compte.
3. Dans la page de navigation, choisissez Account Settings (Paramètres du compte).
4. Choisissez Mettre à jour.
5. Choisissez Revert to account default (Restaurer la valeur par défaut du compte).
6. Dans l'écran de confirmation, choisissez Confirm (Confirmer) pour enregistrer la sélection.

Gestion des paramètres du compte Amazon ECS à l'aide du AWS CLI

Vous pouvez gérer les paramètres de votre compte à l'aide de l'API Amazon ECS, AWS CLI ou SDKs. Les paramètres `fargateFIPSMode` et `dualStackIPv6` les paramètres du `fargateTaskRetirementWaitPeriod` compte ne peuvent être consultés ou modifiés qu'à l'aide de ces outils.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir de AWS CLI SDKs, et de l'API Amazon ECS via les deux IPv4 et. IPv6 Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

Pour plus d'informations sur les actions d'API disponibles pour les définitions de tâches, veuillez consulter Actions relatives à la configuration du compte dans la Référence de l'API Amazon Elastic Container Service (langue française non garantie).

Utilisez l'une des commandes suivantes pour modifier les paramètres de compte par défaut pour tous les utilisateurs ou rôles de votre compte. Ces modifications s'appliquent à l'ensemble du AWS compte, sauf si un utilisateur ou un rôle remplace explicitement ces paramètres pour lui-même.

- put-account-setting-default (AWS CLI)

```
aws ecs put-account-setting-default --name serviceLongArnFormat --value enabled --
region us-east-2
```

Vous pouvez également utiliser cette commande pour modifier d'autres paramètres de compte. Pour ce faire, remplacez le paramètre name par le paramètre de compte correspondant.

- ECSAccountRéglage de l'écriture (AWS Tools for Windows PowerShell)

```
Write-ECSAccountSettingDefault -Name serviceLongArnFormat -Value enabled -Region us-
east-1 -Force
```

Pour modifier les paramètres de compte pour votre compte utilisateur (AWS CLI)

Utilisez l'une des commandes suivantes pour modifier les paramètres de compte pour votre utilisateur . Si vous utilisez ces commandes en tant qu'utilisateur root, ces modifications s'appliquent à l'ensemble du compte AWS , à moins qu'un utilisateur ou un rôle remplace explicitement ces paramètres pour lui-même.

- put-account-setting (AWS CLI)

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled --region us-
east-1
```

Vous pouvez également utiliser cette commande pour modifier d'autres paramètres de compte. Pour ce faire, remplacez le paramètre name par le paramètre de compte correspondant.

- ECSAccountRéglage de l'écriture (AWS Tools for Windows PowerShell)

```
Write-ECSAccountSetting -Name serviceLongArnFormat -Value enabled -Force
```

Pour modifier les paramètres de compte pour un utilisateur ou un rôle spécifique (AWS CLI)

Utilisez l'une des commandes suivantes et spécifiez dans la requête l'ARN d'un utilisateur, d'un rôle ou de l'utilisateur root pour modifier les paramètres de compte pour un utilisateur ou un rôle spécifique.

- put-account-setting (AWS CLI)

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled --principal-
arn arn:aws:iam::aws_account_id:user/principalName --region us-east-1
```

Vous pouvez également utiliser cette commande pour modifier d'autres paramètres de compte. Pour ce faire, remplacez le paramètre `name` par le paramètre de compte correspondant.

- ECSAccountRéglage de l'écriture (AWS Tools for Windows PowerShell)

```
Write-ECSAccountSetting -Name serviceLongArnFormat -Value enabled -PrincipalArn
 arn:aws:iam::aws_account_id:user/principalName -Region us-east-1 -Force
```

Rôles IAM pour Amazon ECS

Un rôle IAM est une identité IAM que vous pouvez créer dans votre compte et qui dispose d'autorisations spécifiques. Dans Amazon ECS, vous pouvez créer des rôles pour accorder des autorisations à des ressources Amazon ECS telles que des conteneurs ou des services.

Les rôles requis par Amazon ECS dépendent de la définition de tâche, du type de lancement et des fonctionnalités que vous utilisez. Utilisez le tableau suivant pour déterminer les rôles IAM dont vous avez besoin pour Amazon ECS.

Rôle	Définition	En cas de besoin	En savoir plus
Rôle d'exécution de tâche	Ce rôle permet à Amazon ECS d'utiliser d'autres AWS services en votre nom.	Votre tâche est hébergée sur AWS Fargateou sur des instances externes et : • extrait une image de conteneur depuis un référentiel privé Amazon ECR. • extrait une image de conteneur d'un référentiel privé Amazon ECR dans un compte différent	Rôle IAM d'exécution de tâche Amazon ECS

Rôle	Définition	En cas de besoin	En savoir plus
		de celui qui exécute la tâche. • envoie les journaux des conteneurs à CloudWatch Logs à l'aide du pilote de `awslogs` journal. Votre tâche est hébergée sur l'une AWS Fargate ou l'autre EC2 instance Amazon et : • utilise l'authent ification du registre privé. • utilise la surveilla nce du temps d'exécution. • la définition de la tâche fait référence à des données sensibles à l'aide des secrets de Secrets Manager ou AWS des paramètres du magasin de paramètres de Systems Manager.	

Rôle	Définition	En cas de besoin	En savoir plus
Rôle de tâche	Ce rôle permet au code de votre application (sur le conteneur) d'utiliser d'autres AWS services.	Votre application accède à d'autres AWS services, tels qu'Amazon S3.	Rôle IAM de la tâche Amazon ECS
Rôle de l'instance de conteneur	Ce rôle permet à vos EC2 instances ou à des instances externes de s'enregistrer auprès du cluster.	Votre tâche est hébergée sur des EC2 instances Amazon ou sur une instance externe.	Rôle IAM d'instance de conteneur Amazon ECS
Rôle dans Amazon ECS Anywhere	Ce rôle permet à vos instances externes d'y accéder AWS APIs.	Votre tâche est hébergée sur des instances externes.	Rôle IAM dans Amazon ECS Anywhere
Rôle de l'infrastructure Amazon ECS pour les équilibreurs de charge	Ce rôle permet à Amazon ECS de gérer les ressources de l'équilibreur de charge dans vos clusters en votre nom pour les blue/green déploiements.	Vous souhaitez utiliser les blue/green déploiements Amazon ECS.	Rôle IAM de l'infrastructure Amazon ECS pour les équilibreurs de charge
CodeDeploy Rôle Amazon ECS	Ce rôle permet CodeDeploy de mettre à jour vos services.	Vous utilisez le type de déploiement CodeDeploy bleu/vert pour déployer des services.	Rôle CodeDeploy IAM d'Amazon ECS
EventBridge Rôle Amazon ECS	Ce rôle permet EventBridge de mettre à jour vos services.	Vous utilisez les EventBridge règles et les objectifs pour planifier vos tâches.	Rôle EventBridge IAM d'Amazon ECS

Rôle	Définition	En cas de besoin	En savoir plus
Rôle de l'infrast ructure Amazon ECS	Ce rôle permet à Amazon ECS de gérer les ressources d'infrastructure de vos clusters.	• Vous souhaitez associer des volumes Amazon EBS à vos tâches Amazon ECS de type Fargate ou de EC2 lancement . Le rôle d'infrast ructure permet à Amazon ECS de gérer les volumes Amazon EBS pour vos tâches. • Vous souhaitez utiliser le protocole TLS (Transport Layer Security) pour chiffrer le trafic entre vos services Amazon ECS Service Connect. • Vous souhaitez créer des groupes cibles VPC Lattice.	Rôle IAM dans l'infrastructure Amazon ECS

Définitions de tâche Amazon ECS

Une définition de tâche est le plan de votre application. Il s'agit d'un fichier texte au format JSON qui décrit les paramètres et un ou plusieurs conteneurs qui forment votre application.

Voici certains des paramètres que vous pouvez spécifier dans une définition de tâche :

- Le type de lancement à utiliser, qui détermine l'infrastructure sur laquelle vos tâches sont hébergées
- L'image Docker à utiliser avec chaque conteneur dans la tâche
- Les ressources d'UC et de mémoire à utiliser avec chaque tâche ou chaque conteneur au sein d'une tâche
- Les exigences en matière de mémoire et de processeur
- Système d'exploitation du conteneur sur lequel la tâche s'exécute
- Le mode réseau Docker à utiliser pour les conteneurs dans votre tâche
- La configuration de journalisation à utiliser pour les tâches
- Si la tâche continue à s'exécuter en cas d'arrêt ou d'échec du conteneur
- La commande que le conteneur exécute au démarrage
- Les volumes de données utilisés avec les conteneurs dans la tâche
- Le rôle IAM que les tâches utilisent

Pour obtenir la liste complète des paramètres de définition de tâche, veuillez consulter Paramètres de définition des tâches Amazon ECS pour le type de lancement Fargate.

Après avoir créé une définition de tâche, vous pouvez l'exécuter en tant que tâche ou service.

- Une tâche est l'instanciation d'une définition de tâche au sein d'un cluster. Après avoir créé une définition de tâche pour votre application dans Amazon ECS, vous pouvez spécifier le nombre de tâches à exécuter sur votre cluster.
- Un service Amazon ECS service exécute et conserve simultanément le nombre souhaité de tâches dans un cluster Amazon ECS. Le principe est le suivant : si l'une de vos tâches échoue ou s'arrête pour une raison quelconque, le planificateur de service d'Amazon ECS service lance une autre instance en fonction de votre définition de tâche. Il procède ainsi pour le remplacer et donc maintenir le nombre de tâches souhaité dans le service.

Rubriques

- États de définition des tâches Amazon ECS
- Architectez votre application pour Amazon ECS
- Création d'une définition de tâche Amazon ECS à l'aide de la console
- Mettre à jour une définition de tâche Amazon ECS à l'aide de la console
- Annulation de l'enregistrement d'une révision de définition de tâche Amazon ECS à l'aide de la console
- Suppression d'une révision de définition de tâche Amazon ECS à l'aide de la console
- Cas d'utilisation de la définition des tâches Amazon ECS
- Paramètres de définition des tâches Amazon ECS pour le type de lancement Fargate
- Paramètres de définition des tâches Amazon ECS pour le type de EC2 lancement
- Modèle de définition de tâche Amazon ECS
- Exemples de définitions de tâches Amazon ECS

États de définition des tâches Amazon ECS

Une définition de tâche change d'état lorsque vous la créez, la désenregistrez ou la supprimez. Vous pouvez consulter l'état de la définition des tâches dans la console ou en utilisant`DescribeTaskDefinition`.

Les états possibles pour une définition de tâche sont les suivants :

ACTIF

La définition d'une tâche est `ACTIVE` après son enregistrement auprès d'Amazon ECS. Les définitions de tâches à l'état `ACTIVE` vous permettent d'exécuter des tâches ou de créer des services.

INACTIVE

Une définition de tâche passe de l'état `ACTIVE` à l'état `INACTIVE` lorsque vous annulez l'enregistrement d'une définition de tâche. Vous pouvez récupérer la définition d'une tâche `INACTIVE` en appelant `DescribeTaskDefinition`. Vous ne pouvez pas exécuter de nouvelles tâches ou créer de nouveaux services avec une définition de tâche à l'état `INACTIVE`. Il n'y a aucun impact sur les services ou les tâches existants.

DELETE_IN_PROGRESS

Une définition de tâche passe de l'état `INACTIVE` à l'état `DELETE_IN_PROGRESS` une fois que vous l'avez soumise pour suppression. Une fois que la définition de tâche est à l'état `DELETE_IN_PROGRESS`, Amazon ECS vérifie régulièrement que la définition de tâche cible n'est référencée par aucune tâche ou déploiement actif, puis supprime définitivement la définition de tâche. Vous ne pouvez pas exécuter de nouvelles tâches ou créer de nouveaux services avec une définition de tâche à l'état `DELETE_IN_PROGRESS`. Une définition de tâche peut être soumise pour suppression à tout moment sans affecter les tâches et services existants.

Les définitions de tâches dont l'état est `DELETE_IN_PROGRESS` peuvent être consultées dans la console et vous pouvez récupérer la définition de tâche en appelant `DescribeTaskDefinition`.

Lorsque vous supprimez toutes les révisions de définition de tâche `INACTIVE`, le nom de la définition de tâche n'est pas affiché dans la console et n'est pas renvoyé dans l'API. Si une révision de définition de tâche est dans `DELETE_IN_PROGRESS` cet état, le nom de la définition de tâche est affiché dans la console et renvoyé dans l'API. Le nom de la définition de tâche est conservé par Amazon ECS et la révision est incrémentée la prochaine fois que vous créerez une définition de tâche portant ce nom.

Si vous avez l' AWS Config habitude de gérer vos définitions de tâches, tous les enregistrements de définitions de tâches vous sont AWS Config facturés. Vous n'êtes facturé que pour la désinscription de la dernière définition de tâche `ACTIVE`. La suppression d'une définition de tâche est gratuite. Pour plus d'informations sur la tarification, consultez Tarification d'AWS Config.

Ressources Amazon ECS pouvant bloquer une suppression

Une demande de suppression de définition de tâche ne sera pas traitée lorsque certaines ressources Amazon ECS dépendent de la révision de la définition de tâche. Les ressources suivantes peuvent empêcher la suppression d'une définition de tâche :

- Tâches autonomes Amazon ECS : la définition de la tâche est requise pour que la tâche reste saine.
- Tâches de service Amazon ECS : la définition de la tâche est requise pour que la tâche reste saine.

- Déploiements de services et ensembles de tâches Amazon ECS : la définition des tâches est requise lorsqu'un événement de dimensionnement est lancé pour un déploiement ou un ensemble de tâches Amazon ECS.

Si votre définition de tâche reste `DELETE_IN_PROGRESS` inchangée, vous pouvez utiliser la console ou le AWS CLI pour identifier, puis arrêter les ressources qui bloquent la suppression de la définition de tâche.

Suppression de la définition de tâche après la suppression de la ressource bloquée

Les règles suivantes s'appliquent une fois que vous avez supprimé les ressources bloquant la suppression de la définition de tâche :

- Tâches Amazon ECS : la suppression de la définition de tâche peut prendre jusqu'à une heure après l'arrêt de la tâche.
- Déploiements de services et ensembles de tâches Amazon ECS : la suppression de la définition de tâche peut prendre jusqu'à 24 heures après la suppression du déploiement ou de l'ensemble de tâches.

Architectez votre application pour Amazon ECS

Vous concevez l'architecture de votre application en créant une définition de tâche pour votre application. La définition de tâche contient les paramètres qui définissent les informations relatives à l'application, notamment :

- Type de lancement à utiliser, qui détermine l'infrastructure sur laquelle vos tâches sont hébergées.

  Lorsque vous utilisez le type de EC2 lancement, vous choisissez également le type d'instance. Pour certains types d'instances, tels que le GPU, vous devez définir des paramètres supplémentaires. Pour de plus amples informations, veuillez consulter Cas d'utilisation de la définition des tâches Amazon ECS.
- L'image du conteneur, qui contient le code de votre application et toutes les dépendances dont le code d'application a besoin pour s'exécuter.
- Mode réseau à utiliser pour les conteneurs de votre tâche.

  Le mode réseau détermine la manière dont votre tâche communique sur le réseau.

Pour les tâches exécutées sur EC2 des instances, il existe plusieurs options, mais nous vous recommandons d'utiliser le mode `awsvpc` réseau. Le mode `awsvpc` réseau simplifie la mise en réseau des conteneurs en vous permettant de mieux contrôler la manière dont vos applications communiquent entre elles et avec les autres services au sein de votre entreprise VPCs.

Pour les tâches exécutées sur Fargate, vous devez utiliser `awsvpc` le mode réseau.

- La configuration de journalisation à utiliser pour vos tâches.
- Tous les volumes de données utilisés avec les conteneurs de la tâche.

Pour obtenir la liste complète des paramètres de définition de tâche, veuillez consulter Paramètres de définition des tâches Amazon ECS pour le type de lancement Fargate.

Suivez ces recommandations lors de la création de vos définitions de tâche :

- Utilisez chaque famille de définitions de tâches pour un seul objectif métier.

  Si vous regroupez plusieurs types de conteneurs d'applications dans la même définition de tâche, vous ne pouvez pas redimensionner ces conteneurs indépendamment. Par exemple, un site Web et une API nécessitent généralement des modèles de mise à l'échelle différents. À mesure que le trafic augmente, il se peut que vous ayez besoin d'un nombre de conteneurs Web différent de celui des conteneurs d'API. Si ces deux conteneurs sont déployés dans la même définition de tâche, chaque tâche exécute le même nombre de conteneurs Web et de conteneurs d'API.
- Associez chaque version d'application à une révision de définition de tâche au sein d'une famille de définitions de tâches.

  Au sein d'une famille de définitions de tâches, chaque révision de définition de tâche représente un point-in-time instantané des paramètres d'une image de conteneur particulière. Cela est similaire à la façon dont le conteneur est un instantané de tous les composants nécessaires pour exécuter une version particulière du code de votre application.

  Créez un one-to-one mappage entre une version du code de l'application, une balise d'image de conteneur et une révision de définition de tâche. Un processus de publication typique implique une validation git qui est transformée en une image de conteneur balisée avec le SHA de validation git. Cette balise d'image de conteneur reçoit ensuite sa propre révision de définition de tâche Amazon ECS. Enfin, le service Amazon ECS est mis à jour pour déployer la nouvelle révision de définition de tâche.
- Utilisez différents rôles IAM pour chaque famille de définitions de tâches.

Définissez chaque définition de tâche avec son propre rôle IAM. Mettez en œuvre cette pratique tout en fournissant à chaque composant métier sa propre famille de définitions de tâches.
En mettant en œuvre ces deux meilleures pratiques, vous pouvez limiter l'accès de chaque service aux ressources de votre AWS compte. Par exemple, vous pouvez autoriser votre service d'authentification à accéder à votre base de données de mots de passe. Dans le même temps, vous pouvez vous assurer que seul votre service de commande a accès aux informations de paiement par carte de crédit.

Bonnes pratiques pour les images de conteneurs Amazon ECS

Une image de conteneur est un ensemble d'instructions expliquant comment créer le conteneur. Une image de conteneur contient le code de votre application et toutes les dépendances dont le code d'application a besoin pour s'exécuter. Les dépendances des applications incluent les packages de code source sur lesquels repose le code de votre application, un environnement d'exécution de langage pour les langages interprétés et les packages binaires sur lesquels repose votre code lié dynamiquement.

Suivez ces directives lors de la conception et de la création de vos images de conteneur :

- Complétez vos images de conteneur en stockant toutes les dépendances de l'application sous forme de fichiers statiques à l'intérieur de l'image de conteneur.

  Si vous modifiez quelque chose dans l'image du conteneur, créez une nouvelle image du conteneur avec les modifications.
- Exécutez un seul processus de candidature dans un conteneur.

  La durée de vie du conteneur est aussi longue que le processus de l'application. Amazon ECS remplace les processus bloqués et détermine où lancer le processus de remplacement. Une image complète rend le déploiement global plus résilient.
- Gérez votre application`SIGTERM`.

  Lorsqu'Amazon ECS arrête une tâche, il envoie d'abord un signal SIGTERM à la tâche pour informer l'application qu'elle doit terminer et s'arrêter. Amazon ECS envoie ensuite un `SIGKILL` message. Lorsque les applications ignorent le`SIGTERM`, le service Amazon ECS doit attendre avant d'envoyer le `SIGKILL` signal pour mettre fin au processus.

Vous devez déterminer le temps nécessaire à votre application pour terminer son travail et vous assurer que vos applications gèrent le `SIGTERM` signal. La gestion du signal par l'application doit empêcher l'application d'effectuer de nouveaux travaux et de terminer le travail en cours, ou enregistrer le travail inachevé dans un espace de stockage en dehors de la tâche lorsque le travail prend trop de temps à terminer.

- Configurez des applications conteneurisées pour écrire des journaux dans `stdout` et `stderr`.

  Le découplage de la gestion des journaux du code de votre application vous donne la flexibilité d'ajuster la gestion des journaux au niveau de l'infrastructure. La modification de votre système de journalisation en est un exemple. Au lieu de modifier vos services, de créer et de déployer une nouvelle image de conteneur, vous pouvez ajuster les paramètres.

- Utilisez des balises pour gérer les versions des images de vos conteneurs.

  Les images de conteneurs sont stockées dans un registre de conteneurs. Chaque image d'un registre est identifiée par une balise. Une balise est appelée `latest`. Cette balise est dirigée vers la dernière version de l'image du conteneur de l'application, comme `HEAD` dans un référentiel git. Nous vous recommandons d'utiliser uniquement la balise `latest` à des fins de test. Il est recommandé de baliser les images des conteneurs avec une balise unique pour chaque compilation. Nous vous recommandons de baliser vos images en utilisant le git SHA pour la validation git qui a été utilisée pour créer l'image.

  Vous n'avez pas besoin de créer une image de conteneur pour chaque validation. Toutefois, nous vous recommandons de créer une nouvelle image de conteneur chaque fois que vous publiez une validation de code spécifique dans l'environnement de production. Nous vous recommandons également de baliser l'image avec une balise correspondant à la validation git du code contenu dans l'image. Si vous avez balisé l'image avec la validation git, vous pouvez trouver plus rapidement la version du code exécutée par l'image.

  Nous vous recommandons également d'activer les balises d'image immuables dans Amazon Elastic Container Registry. Avec ce paramètre, vous ne pouvez pas modifier l'image du conteneur vers laquelle pointe une balise. Amazon ECR impose plutôt qu'une nouvelle image soit téléchargée vers un nouveau tag. Pour plus d'informations, veuillez consulter Mutabilité d'une étiquette d'image dans le Guide de l'utilisateur Amazon ECR.

Lorsque vous concevez votre application pour qu'elle s'exécute AWS Fargate, vous devez choisir entre déployer plusieurs conteneurs dans la même définition de tâche ou déployer des conteneurs

séparément dans plusieurs définitions de tâches. Si les conditions suivantes sont requises, nous vous recommandons de déployer plusieurs conteneurs dans une définition de tâche unique :

- Vos conteneurs partagent le même cycle de vie (autrement dit, ils sont lancés et résiliés conjointement).
- Vos conteneurs doivent être exécutés sur le même hôte sous-jacent (autrement dit, un conteneur référence l'autre sur un port localhost).
- Vos conteneurs partagent des ressources.
- Vos conteneurs partagent des volumes de données.

Si ces conditions suivantes ne sont pas requises, nous vous recommandons de déployer des conteneurs distincts dans plusieurs définitions de tâche. Cela vous permet de dimensionner, d'approvisionner et de déprovisionner les conteneurs séparément.

Bonnes pratiques relatives à la taille des tâches Amazon ECS

La taille de vos conteneurs et de vos tâches est essentielle à la mise à l'échelle et à la planification des capacités. Dans Amazon ECS, le processeur et la mémoire sont deux indicateurs de ressources utilisés pour déterminer la capacité. Le processeur est mesuré en unités de 1/1024 d'un vCPU complet (1024 unités étant égales à 1 vCPU entier). La mémoire est mesurée en mégaoctets. Dans la définition de votre tâche, vous pouvez configurer les réservations et les limites des ressources.

Lorsque vous configurez une réservation, vous définissez la quantité minimale de ressources requise par une tâche. Votre tâche reçoit au moins le montant des ressources demandées. Votre application peut utiliser plus de processeur ou de mémoire que la réservation que vous déclarez. Toutefois, cela est soumis aux limites que vous avez également déclarées. Le fait d'utiliser un montant supérieur au montant de la réservation est appelé « bursting ». Dans Amazon ECS, les réservations sont garanties. Par exemple, si vous utilisez des EC2 instances Amazon pour fournir de la capacité, Amazon ECS ne place aucune tâche sur une instance où la réservation ne peut pas être exécutée.

Une limite est la quantité maximale d'unités de processeur ou de mémoire que votre conteneur ou votre tâche peut utiliser. Toute tentative d'utilisation d'un processeur supérieur à cette limite entraîne un ralentissement. Toute tentative d'utilisation de plus de mémoire entraîne l'arrêt de votre conteneur.

Le choix de ces valeurs peut s'avérer difficile. En effet, les valeurs les mieux adaptées à votre application dépendent dans une large mesure des besoins en ressources de votre application. Le test de charge de votre application est la clé d'une planification réussie des besoins en ressources et d'une meilleure compréhension des exigences de votre application.

Demandes apatrides

Pour les applications sans état qui évoluent horizontalement, telles qu'une application située derrière un équilibreur de charge, nous vous recommandons de déterminer d'abord la quantité de mémoire consommée par votre application lorsqu'elle traite des demandes. Pour ce faire, vous pouvez utiliser des outils traditionnels tels que `ps` ou `top` des solutions de surveillance telles que CloudWatch Container Insights.

Lorsque vous déterminez une réservation de processeur, réfléchissez à la manière dont vous souhaitez adapter votre application aux besoins de votre entreprise. Vous pouvez utiliser des réserves de processeur plus petites, telles que 256 unités de processeur (ou 1/4 de vCPU), pour effectuer une mise à l'échelle précise afin de minimiser les coûts. Cependant, ils risquent de ne pas évoluer assez rapidement pour répondre à des pics de demande importants. Vous pouvez utiliser des réservations de processeur plus importantes pour augmenter les volumes entrant et sortant plus rapidement et ainsi répondre plus rapidement aux pics de demande. Cependant, les réservations de processeurs plus importantes sont plus coûteuses.

Autres applications

Pour les applications qui ne sont pas évolutives horizontalement, telles que les serveurs de base de données ou les serveurs de base de données, la capacité disponible et les coûts constituent vos considérations les plus importantes. Vous devez choisir la quantité de mémoire et de processeur en fonction des tests de charge indiquant que vous devez desservir le trafic pour atteindre votre objectif de niveau de service. Amazon ECS garantit que l'application est placée sur un hôte doté d'une capacité adéquate.

Options de mise en réseau des tâches Amazon ECS pour le type de EC2 lancement

Le comportement réseau des tâches Amazon ECS hébergées sur des EC2 instances Amazon dépend du mode réseau défini dans la définition de la tâche. Nous vous recommandons d'utiliser le mode réseau `awsvpc`, à moins que vous n'ayez besoin d'utiliser un mode réseau différent.

Les modes réseau disponibles sont les suivants.

Mode réseau	Conteneurs Linux sur EC2	Conteneurs Windows activés EC2	Description
`awsvpc`	Oui	Oui	La tâche se voit attribuer sa propre interface Elastic Network (ENI) et une IPv4 adresse privée principale. Cela confère à la tâche les mêmes propriétés réseau que les EC2 instances Amazon.
`bridge`	Oui	Non	La tâche utilise le réseau virtuel intégré de Docker sous Linux, qui s'exécute dans chaque EC2 instance Amazon hébergeant la tâche. Le réseau virtuel intégré sous Linux utilise le pilote réseau Docker `bridge`. Ceci est le mode réseau par défaut sous Linux si aucun mode réseau n'est spécifié dans la définition de la tâche.
`host`	Oui	Non	La tâche utilise le réseau de l'hôte qui contourne le réseau virtuel intégré de Docker en mappant les ports des conteneurs directement à l'ENI de l' EC2 instance Amazon qui héberge la tâche. Les mappages de ports dynamiques ne peuvent pas être utilisés dans ce mode réseau. Dans une définition de tâche qui utilise ce mode, un conteneur doit spécifier un numéro `hostPort` spécifique. Un numéro de port sur un hôte ne peut pas être utilisé par plusieurs tâches. Par conséquent, vous ne pouvez pas exécuter plusieurs tâches ayant la même définition de tâche sur une seule EC2 instance Amazon.
`none`	Oui	Non	La tâche n'a pas de connectivité réseau externe.

Mode réseau	Conteneurs Linux sur EC2	Conteneurs Windows activés EC2	Description
`default`	Non	Oui	La tâche utilise le réseau virtuel intégré de Docker sous Windows, qui s'exécute dans chaque EC2 instance Amazon hébergeant la tâche. Le réseau virtuel intégré sous Windows utilise le pilote réseau Docker `nat`. Ceci est le mode réseau par défaut sous Windows si aucun mode réseau n'est spécifié dans la définition de la tâche.

Pour plus d'informations sur la mise en réseau Docker sous Linux, consultez Présentation de la mise en réseau dans la Documentation Docker.

Pour plus d'informations sur la mise en réseau Docker sous Windows, consultez Mise en réseau de conteneurs Windows dans la Documentation Microsoft sur les conteneurs Windows.

Allouer une interface réseau pour une tâche Amazon ECS

Les fonctionnalités de mise en réseau des tâches fournies par le mode `awsvpc` réseau confèrent aux tâches Amazon ECS les mêmes propriétés réseau que EC2 les instances Amazon. L'utilisation du mode `awsvpc` réseau simplifie la mise en réseau des conteneurs, car vous avez un meilleur contrôle sur la façon dont vos applications communiquent entre elles et avec les autres services au sein de votre entreprise VPCs. Le mode `awsvpc` réseau renforce également la sécurité de vos conteneurs en vous permettant d'utiliser des groupes de sécurité et des outils de surveillance réseau à un niveau plus détaillé dans le cadre de vos tâches. Vous pouvez également utiliser d'autres fonctionnalités EC2 réseau d'Amazon, telles que les journaux de flux VPC pour surveiller le trafic en provenance et à destination de vos tâches. De plus, les conteneurs qui appartiennent à la même tâche peuvent communiquer via l'interface `localhost`.

La task elastic network interface (ENI) est une fonctionnalité entièrement gérée d'Amazon ECS. Amazon ECS crée l'ENI et l'attache à l' EC2 instance Amazon hôte avec le groupe de sécurité spécifié. La tâche envoie et reçoit du trafic réseau via l'ENI de la même manière que les EC2 instances Amazon le font avec leurs interfaces réseau principales. Une IPv4 adresse privée est attribuée par défaut à chaque tâche ENI. Si votre VPC est activé pour le mode double pile et que

vous utilisez un sous-réseau avec un bloc IPv6 CIDR, la tâche ENI recevra également une adresse. IPv6 Chaque tâche ne peut avoir qu'une seule ENI.

Ils ENIs sont visibles dans la EC2 console Amazon de votre compte. Votre compte ne peut pas détacher ou modifier le ENIs. L'objectif est d'empêcher la suppression accidentelle de toute ENI associée à une tâche en cours d'exécution. Vous pouvez consulter les informations de pièce jointe ENI pour les tâches dans la console Amazon ECS ou lors du fonctionnement de l'DescribeTasksAPI. Lorsque la tâche s'arrête ou si le service est réduit, l'ENI de la tâche est détachée et supprimée.

Lorsque vous avez besoin d'une densité ENI accrue, utilisez les paramètres du `awsvpcTrunking` compte. Amazon ECS crée et attache également une interface réseau « tronc » pour votre instance de conteneur. Le réseau « de jonction » est entièrement géré par Amazon ECS. L'ENI « de jonction » est supprimée lorsque vous résiliez votre instance de conteneur ou lorsque vous annulez son enregistrement dans le cluster Amazon ECS. Pour plus d'informations sur le paramétrage du `awsvpcTrunking` compte, consultezPrérequis.

Vous le spécifiez `awsvpc` dans le `networkMode` paramètre de la définition de la tâche. Pour de plus amples informations, veuillez consulter Mode réseau.

Ensuite, lorsque vous exécutez une tâche ou créez un service, utilisez le `networkConfiguration` paramètre qui inclut un ou plusieurs sous-réseaux dans lesquels placer vos tâches et un ou plusieurs groupes de sécurité à associer à un ENI. Pour de plus amples informations, veuillez consulter Configuration réseau. Les tâches sont placées sur des EC2 instances Amazon compatibles situées dans les mêmes zones de disponibilité que ces sous-réseaux, et les groupes de sécurité spécifiés sont associés à l'ENI configuré pour la tâche.

Considérations relatives à Linux

Tenez compte des éléments suivants lorsque vous utilisez le système d'exploitation Linux.

- Si vous utilisez une instance p5.48xlarge en `awsvpc` mode, vous ne pouvez pas exécuter plus d'une tâche sur l'instance.
- Les tâches et les services qui utilisent le mode `awsvpc` réseau nécessitent le rôle lié au service Amazon ECS pour fournir à Amazon ECS les autorisations nécessaires pour passer des appels vers d'autres AWS services en votre nom. Ce rôle est généré automatiquement lorsque vous créez un cluster, ou si vous créez ou mettez à jour un service dans la AWS Management Console. Pour de plus amples informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS. Vous pouvez également créer le rôle lié à un service à l'aide de la commande suivante : AWS CLI

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- Votre instance Amazon EC2 Linux nécessite une version `1.15.0` ou une version ultérieure de l'agent de conteneur pour exécuter des tâches utilisant le mode `awsvpc` réseau. Si vous utilisez l'AMI Linux optimisée pour Amazon ECS, votre instance doit également au moins disposer de la version `1.15.0-4` du package `ecs-init`.
- Amazon ECS remplit le nom d'hôte d'une tâche avec un nom d'hôte DNS (interne) fourni par Amazon lorsque les options `enableDnsHostnames` et `enableDnsSupport` sont toutes deux activées sur votre VPC. Si ces options ne sont pas activées, le nom d'hôte DNS de la tâche est un nom d'hôte aléatoire. Pour plus d'informations sur les paramètres DNS d'un VPC, veuillez consulter la rubrique Utilisation de DNS avec votre VPC dans le Guide de l'utilisateur Amazon VPC.
- Chaque tâche Amazon ECS qui utilise le mode `awsvpc` réseau reçoit sa propre interface Elastic Network (ENI), qui est attachée à l' EC2 instance Amazon qui l'héberge. Il existe un quota par défaut pour le nombre d'interfaces réseau pouvant être associées à une instance Amazon EC2 Linux. L'interface réseau principale est considérée comme une seule pour ce quota. Par exemple, par défaut, une `c5.large` instance peut n'en avoir ENIs que trois qui peuvent être attachées. L'interface réseau principale de l'instance est considérée comme une interface réseau. Vous pouvez en attacher deux autres ENIs à l'instance. Dans la mesure où chaque tâche utilisant le mode réseau `awsvpc` nécessite une ENI, vous ne pouvez généralement exécuter que deux tâches sur ce type d'instance. Pour plus d'informations sur les limites ENI par défaut pour chaque type d'instance, consultez la section Adresses IP par interface réseau et par type d'instance dans le guide de EC2 l'utilisateur Amazon.
- Amazon ECS prend en charge le lancement d'instances Amazon EC2 Linux qui utilisent des types d'instances pris en charge avec une densité ENI accrue. Lorsque vous activez les paramètres du `awsvpcTrunking` compte et que vous enregistrez des instances Amazon EC2 Linux qui utilisent ces types d'instances dans votre cluster, le quota ENI de ces instances est plus élevé. L'utilisation de ces instances avec ce quota plus élevé signifie que vous pouvez placer davantage de tâches sur chaque instance Amazon EC2 Linux. Pour utiliser la densité ENI accrue avec la fonctionnalité de jonction, votre EC2 instance Amazon doit utiliser la version `1.28.1` ou une version ultérieure de l'agent de conteneur. Si vous utilisez une AMI optimisée pour Amazon ECS, votre instance doit également au moins disposer de la version `1.28.1-2` du package `ecs-init`. Pour en savoir plus sur l'acceptation du paramètre de compte `awsvpcTrunking`, consultez Accédez aux fonctionnalités d'Amazon ECS avec les paramètres du compte. Pour en savoir plus sur la jonction ENI, consultez Augmenter les interfaces réseau des instances de conteneur Linux Amazon ECS.

- Lorsque vous hébergez des tâches qui utilisent le mode `awsvpc` réseau sur des instances Amazon EC2 Linux, aucune adresse IP publique ENIs n'est attribuée à votre tâche. Pour accéder à Internet, les tâches doivent être lancées dans un sous-réseau privé configuré pour utiliser une passerelle NAT. Pour plus d'informations, veuillez consulter NAT Gateways (Passerelles NAT) dans le Guide de l'utilisateur Amazon VPC. L'accès entrant au réseau doit se faire depuis un VPC avec l'adresse IP privée ou doit transiter par un équilibreur de charge au sein du VPC. Les tâches lancées dans les sous-réseaux publics n'ont pas accès à Internet.
- Amazon ECS reconnaît uniquement ENIs ce qu'il attache à vos instances Amazon EC2 Linux. Si vous vous connectez manuellement ENIs à vos instances, Amazon ECS peut tenter d'ajouter une tâche à une instance qui ne dispose pas de suffisamment d'adaptateurs réseau. Cela peut entraîner l'expiration de la tâche et le passage à un état de déprovisionnement, puis à un état d'arrêt. Nous vous recommandons de ne pas vous connecter manuellement ENIs à vos instances.
- Les instances Amazon EC2 Linux doivent être enregistrées de manière `ecs.capability.task-eni` à pouvoir être prises en compte pour le placement de tâches en mode `awsvpc` réseau. Les instances exécutant la version `1.15.0-4` ou une version ultérieure d'`ecs-init` sont automatiquement enregistrées avec cet attribut.
- Les ENIs fichiers créés et attachés à vos instances Amazon EC2 Linux ne peuvent pas être détachés manuellement ni modifiés par votre compte. L'objectif est d'empêcher la suppression accidentelle de toute ENI associée à une tâche en cours d'exécution. Pour libérer le ENIs pour une tâche, arrêtez la tâche.
- Il existe une limite de 16 sous-réseaux et 5 groupes de sécurité pouvant être spécifiés dans `awsVpcConfiguration` lors de l'exécution d'une tâche ou de la création d'un service qui utilise le mode réseau `awsvpc`. Pour plus d'informations, consultez le AwsVpcConfigurationmanuel Amazon Elastic Container Service API Reference.
- Lorsqu'une tâche est démarrée avec le mode réseau `awsvpc`, l'agent de conteneur Amazon ECS crée un conteneur `pause` supplémentaire pour chaque tâche avant de démarrer les conteneurs dans la définition de tâche. Il configure ensuite l'espace de noms réseau du `pause` conteneur en exécutant les plugins amazon-ecs-cni-plugins CNI. L'agent démarre ensuite le reste des conteneurs dans la tâche afin qu'ils partagent la pile réseau du conteneur `pause`. Cela signifie que tous les conteneurs d'une tâche peuvent être adressés par les adresses IP de l'ENI et qu'ils peuvent communiquer entre eux via l'interface `localhost`.
- Les services avec des tâches qui utilisent le mode réseau `awsvpc` prennent uniquement en charge Application Load Balancer et Network Load Balancer. Lorsque vous créez des groupes cibles pour ces services, vous devez choisir `ip` comme type de cible. N'utilisez pas `instance`. Cela est dû au fait que les tâches qui utilisent le mode `awsvpc` réseau sont associées à une ENI, et non

à une instance Amazon EC2 Linux. Pour de plus amples informations, veuillez consulter Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS.

- Si votre VPC est mis à jour pour modifier le jeu d'options DHCP qu'il utilise, vous ne pouvez pas appliquer ces modifications aux tâches existantes. Commencez de nouvelles tâches en leur appliquant ces modifications, vérifiez qu'elles fonctionnent correctement, puis arrêtez les tâches existantes afin de modifier ces configurations réseau en toute sécurité.

Considérations relatives à Windows

Voici quelques points à prendre en compte lorsque vous utilisez le système d'exploitation Windows :

- Les instances de conteneur utilisant l'AMI Windows Server 2016 optimisée pour Amazon ECS ne peuvent pas héberger des tâches qui utilisent le mode réseau `awsvpc`. Si vous avez un cluster contenant Windows Server 2016 optimisé pour Amazon ECS AMIs et Windows AMIs prenant en charge le mode `awsvpc` réseau, les tâches utilisant le mode `awsvpc` réseau ne sont pas lancées sur les instances de Windows 2016 Server. Elles sont plutôt lancées sur des instances qui prennent en charge le mode réseau `awsvpc`.
- Votre instance Amazon EC2 Windows nécessite une version `1.57.1` ou une version ultérieure de l'agent de conteneur pour utiliser CloudWatch les métriques pour les conteneurs Windows qui utilisent le mode `awsvpc` réseau.
- Les tâches et les services qui utilisent le mode `awsvpc` réseau nécessitent le rôle lié au service Amazon ECS pour fournir à Amazon ECS les autorisations nécessaires pour passer des appels vers d'autres AWS services en votre nom. Ce rôle est généré automatiquement lorsque vous créez un cluster, ou si vous créez ou mettez à jour un service dans AWS Management Console. Pour de plus amples informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS. Vous pouvez également créer le rôle lié à un service à l'aide de la commande suivante AWS CLI .

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- Votre instance Amazon EC2 Windows nécessite une version `1.54.0` ou une version ultérieure de l'agent de conteneur pour exécuter des tâches utilisant le mode `awsvpc` réseau. Lorsque vous amorcez l'instance, vous devez configurer les options requises pour le mode réseau `awsvpc`. Pour de plus amples informations, veuillez consulter the section called “Démarrage des instances de conteneur”.
- Amazon ECS remplit le nom d'hôte d'une tâche avec un nom d'hôte DNS (interne) fourni par Amazon lorsque les options `enableDnsHostnames` et `enableDnsSupport` sont toutes deux

activées sur votre VPC. Si ces options ne sont pas activées, le nom d'hôte DNS de la tâche est un nom d'hôte aléatoire. Pour plus d'informations sur les paramètres DNS d'un VPC, veuillez consulter la rubrique Utilisation de DNS avec votre VPC dans le Guide de l'utilisateur Amazon VPC.

- Chaque tâche Amazon ECS qui utilise le mode `awsvpc` réseau reçoit sa propre interface Elastic Network (ENI), qui est attachée à l'instance Amazon EC2 Windows qui l'héberge. Il existe un quota par défaut pour le nombre d'interfaces réseau pouvant être associées à une instance Amazon EC2 Windows. L'interface réseau principale est considérée comme une seule pour ce quota. Par exemple, par défaut, une `c5.large` instance peut n'avoir que trois pièces ENIs associées. L'interface réseau principale de l'instance compte parmi celles-ci. Vous pouvez en attacher deux autres ENIs à l'instance. Dans la mesure où chaque tâche utilisant le mode réseau `awsvpc` nécessite une ENI, vous ne pouvez généralement exécuter que deux tâches sur ce type d'instance. Pour plus d'informations sur les limites ENI par défaut pour chaque type d'instance, consultez la section Adresses IP par interface réseau et par type d'instance dans le guide de EC2 l'utilisateur Amazon.
- Lorsque vous hébergez des tâches utilisant le mode `awsvpc` réseau sur des instances Amazon EC2 Windows, aucune adresse IP publique ENIs n'est attribuée à votre tâche. Pour accéder à Internet, lancez les tâches dans un sous-réseau privé configuré pour utiliser une passerelle NAT. Pour plus d'informations, veuillez consulter NAT Gateways (Passerelles NAT) dans le Guide de l'utilisateur Amazon VPC. L'accès entrant au réseau doit se faire depuis le VPC avec l'adresse IP privée ou doit transiter par un équilibreur de charge au sein du VPC. Les tâches lancées dans les sous-réseaux publics n'ont pas accès à Internet.
- Amazon ECS reconnaît uniquement ENIs ce qu'il a attaché à votre instance Amazon EC2 Windows. Si vous vous connectez manuellement ENIs à vos instances, Amazon ECS peut tenter d'ajouter une tâche à une instance qui ne dispose pas de suffisamment d'adaptateurs réseau. Cela peut entraîner l'expiration de la tâche et le passage à un état de déprovisionnement, puis à un état d'arrêt. Nous vous recommandons de ne pas vous connecter manuellement ENIs à vos instances.
- Les instances Amazon EC2 Windows doivent être enregistrées de manière `ecs.capability.task-eni` à pouvoir être prises en compte pour le placement de tâches en mode `awsvpc` réseau.
- Vous ne pouvez pas modifier ou détacher manuellement ENIs ceux qui sont créés et attachés à vos instances Amazon EC2 Windows. Cela vous évite de supprimer accidentellement une ENI associée à une tâche en cours d'exécution. Pour libérer le ENIs pour une tâche, arrêtez la tâche.
- Vous pouvez uniquement spécifier jusqu'à 16 sous-réseaux et 5 groupes de sécurité dans `awsVpcConfiguration`, lorsque vous exécutez une tâche ou créez un service qui utilise le mode

réseau `awsvpc`. Pour plus d'informations, consultez le AwsVpcConfigurationmanuel Amazon Elastic Container Service API Reference.

- Lorsqu'une tâche est démarrée avec le mode réseau `awsvpc`, l'agent de conteneur Amazon ECS crée un conteneur `pause` supplémentaire pour chaque tâche avant de démarrer les conteneurs dans la définition de tâche. Il configure ensuite l'espace de noms réseau du `pause` conteneur en exécutant les plugins amazon-ecs-cni-plugins CNI. L'agent démarre ensuite le reste des conteneurs dans la tâche afin qu'ils partagent la pile réseau du conteneur `pause`. Cela signifie que tous les conteneurs d'une tâche peuvent être adressés par les adresses IP de l'ENI et qu'ils peuvent communiquer entre eux via l'interface `localhost`.
- Les services avec des tâches qui utilisent le mode réseau `awsvpc` prennent uniquement en charge Application Load Balancer et Network Load Balancer. Lorsque vous créez des groupes cibles pour ces services, vous devez choisir le type de cible `ip`, et non `instance`. Cela est dû au fait que les tâches qui utilisent le mode `awsvpc` réseau sont associées à une ENI, et non à une instance Amazon EC2 Windows. Pour de plus amples informations, veuillez consulter Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS.
- Si votre VPC est mis à jour pour modifier le jeu d'options DHCP qu'il utilise, vous ne pouvez pas appliquer ces modifications aux tâches existantes. Commencez de nouvelles tâches en leur appliquant ces modifications, vérifiez qu'elles fonctionnent correctement, puis arrêtez les tâches existantes afin de modifier ces configurations réseau en toute sécurité.
- Les éléments suivants ne sont pas pris en charge lorsque vous utilisez le mode `awsvpc` réseau dans une configuration EC2 Windows :
  - Configuration à double pile
  - IPv6
  - Jonction ENI

Utilisation d'un VPC en mode double pile

Lorsque vous utilisez un VPC en mode double pile, vos tâches peuvent communiquer par ou IPv6 par IPv4 les deux. IPv4 et IPv6 les adresses sont indépendantes les unes des autres. Vous devez donc configurer le routage et la sécurité dans votre VPC séparément pour IPv4 et. IPv6 Pour plus d'informations sur la configuration de votre VPC pour le mode double pile, consultez la section Migration vers le guide de l'utilisateur IPv6 Amazon VPC.

Si vous avez configuré votre VPC avec une passerelle Internet ou une passerelle Internet sortante uniquement, vous pouvez utiliser votre VPC en mode double pile. Ainsi, les tâches auxquelles une IPv6 adresse est attribuée peuvent accéder à Internet via une passerelle Internet ou une passerelle

Internet de sortie uniquement. Les passerelles NAT sont facultatives. Pour plus d'informations, veuillez consulter les rubriques Passerelles Internet et Passerelle Internet en sortie uniquement dans le Guide de l'utilisateur Amazon VPC.

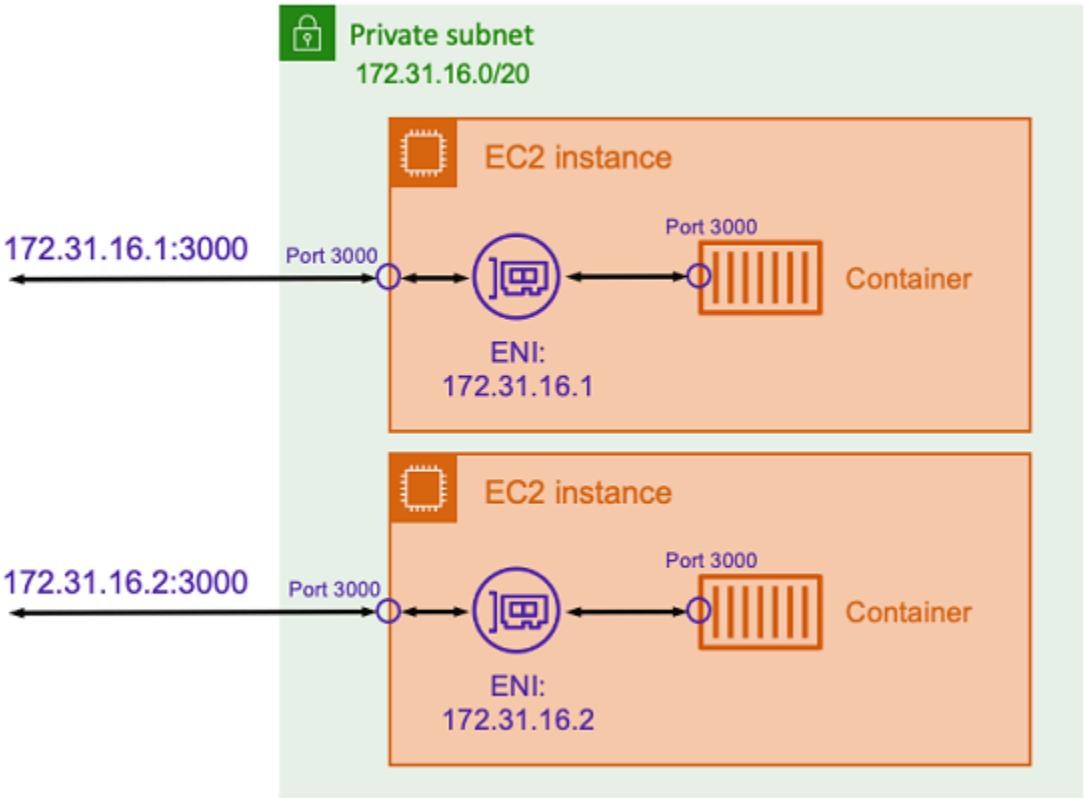
Une IPv6 adresse est attribuée aux tâches Amazon ECS si les conditions suivantes sont remplies :

- L'instance Amazon EC2 Linux qui héberge la tâche utilise la version `1.45.0` ou une version ultérieure de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de l'agent utilisée par votre instance et la mise à jour si nécessaire, veuillez consulter Mise à jour de l'agent de conteneur Amazon ECS.
- Le paramètre de compte `dualStackIPv6` est activé. Pour de plus amples informations, veuillez consulter Accédez aux fonctionnalités d'Amazon ECS avec les paramètres du compte.
- Votre tâche utilise le mode réseau `awsvpc`.
- Votre VPC et votre sous-réseau sont configurés pour. IPv6 La configuration inclut les interfaces réseau créées dans le sous-réseau spécifié. Pour plus d'informations sur la configuration de votre VPC pour le mode double pile, consultez la section Migration vers IPv6 et modification de l'attribut d'IPv6 adressage de votre sous-réseau dans le guide de l'utilisateur Amazon VPC.

Mappez les ports de conteneur Amazon ECS à l'interface réseau de l' EC2 instance

Le mode `host` réseau n'est pris en charge que pour les tâches Amazon ECS hébergées sur EC2 des instances Amazon. Il n'est pas pris en charge lors de l'utilisation d'Amazon ECS sur Fargate.

Le mode réseau `host` est le mode réseau le plus basique pris en charge dans Amazon ECS. En mode hôte, la mise en réseau du conteneur est liée directement à l'hôte sous-jacent qui exécute le conteneur.



Supposons que vous exécutez un conteneur Node.js avec une application Express qui écoute sur un port `3000` similaire à celui illustré dans le schéma précédent. Lorsque le mode `host` réseau est utilisé, le conteneur reçoit le trafic sur le port 3000 en utilisant l'adresse IP de l' EC2 instance Amazon hôte sous-jacente. Nous vous recommandons de ne pas utiliser ce mode.

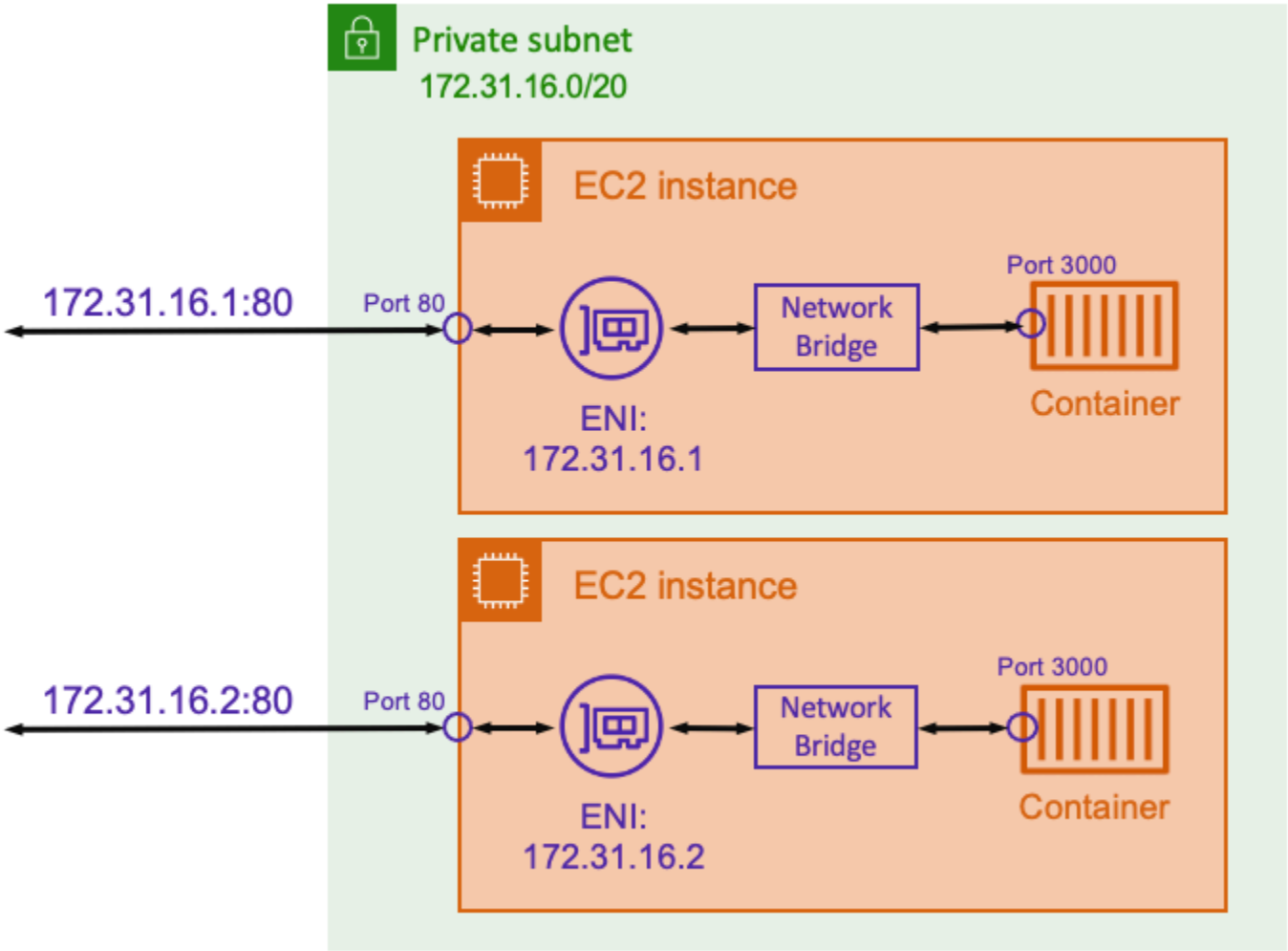
L'utilisation de ce mode réseau présente des inconvénients importants. Vous ne pouvez exécuter qu'une seule instanciation d'une tâche sur chaque hôte. Cela est dû au fait que seule la première tâche peut se lier au port requis sur l' EC2 instance Amazon. Il est également impossible de remapper un port de conteneur lorsqu'il utilise le mode réseau `host`. Par exemple, si une application doit écouter un numéro de port spécifique, vous ne pouvez pas remapper le numéro de port directement. Vous devez plutôt gérer les conflits de ports en modifiant la configuration de l'application.

L'utilisation du mode réseau `host` a également des implications en termes de sécurité. Ce mode permet aux conteneurs de se faire passer pour l'hôte et de se connecter à des services réseau privés en boucle sur l'hôte.

Utiliser le réseau virtuel de Docker pour les tâches Amazon ECS Linux

Le mode `bridge` réseau n'est pris en charge que pour les tâches Amazon ECS hébergées sur EC2 des instances Amazon.

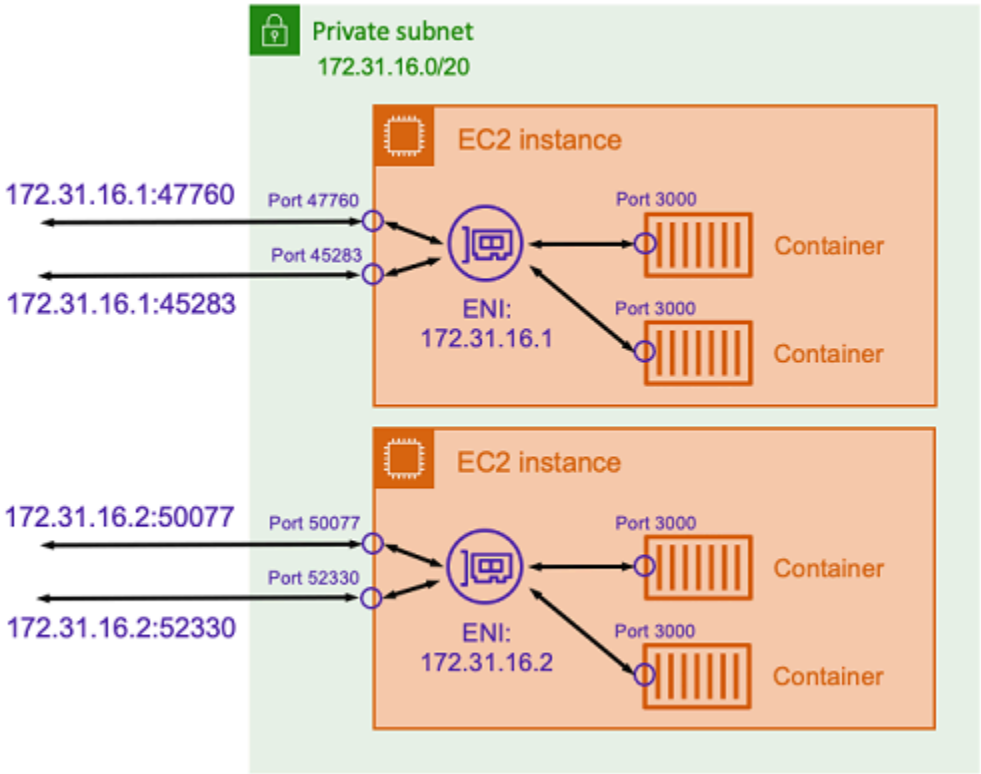
Avec le mode `bridge`, vous utilisez un pont réseau virtuel pour créer une couche entre l'hôte et la mise en réseau du conteneur. De cette façon, vous pouvez créer des mappages de ports qui remappent un port hôte à un port de conteneur. Les mappages peuvent être statiques ou dynamiques.



Avec un mappage statique des ports, vous pouvez définir explicitement le port hôte que vous souhaitez mapper à un port de conteneur. À l'aide de l'exemple ci-dessus, le port `80` de l'hôte est mappé au port `3000` du conteneur. Pour communiquer avec l'application conteneurisée, vous envoyez le trafic au port `80` vers l'adresse IP de l' EC2 instance Amazon. Du point de vue de l'application conteneurisée, le trafic entrant est visible sur le port `3000`.

Si vous souhaitez uniquement modifier le port de trafic, les mappages statiques de ports conviennent. Cependant, cela présente toujours le même inconvénient que l'utilisation du mode réseau `host`. Vous ne pouvez exécuter qu'une seule instanciation d'une tâche sur chaque hôte. Cela est dû au fait qu'un mappage statique de port ne permet de mapper qu'un seul conteneur sur le port 80.

Pour résoudre ce problème, pensez à utiliser le mode réseau `bridge` avec un mappage dynamique des ports, comme indiqué dans le schéma suivant.



En ne spécifiant pas de port hôte dans le mappage des ports, Docker peut choisir un port aléatoire et inutilisé dans la plage de ports éphémères et l'attribuer comme port hôte public pour le conteneur. Par exemple, l'application Node.js qui écoute `3000` le port du conteneur peut se voir attribuer un numéro de port aléatoire élevé, par exemple `47760` sur l' EC2 hôte Amazon. Cela signifie que vous pouvez exécuter plusieurs copies de ce conteneur sur l'hôte. De plus, chaque conteneur peut se voir attribuer son propre port sur l'hôte. Chaque copie du conteneur reçoit du trafic au port `3000`. Toutefois, les clients qui envoient du trafic vers ces conteneurs utilisent les ports hôtes assignés de manière aléatoire.

Amazon ECS vous aide à suivre les ports assignés de manière aléatoire pour chaque tâche. Pour ce faire, il met automatiquement à jour les groupes cibles de l'équilibreur de charge et la découverte des

AWS Cloud Map services afin d'obtenir la liste des adresses IP et des ports des tâches. Cela facilite l'utilisation des services fonctionnant en mode `bridge` utilisant des ports dynamiques.

Cependant, l'un des inconvénients de l'utilisation du mode réseau `bridge` est qu'il est difficile de verrouiller les communications de service à service. Comme les services peuvent être affectés à n'importe quel port aléatoire et inutilisé, il est nécessaire d'ouvrir de larges plages de ports entre les hôtes. Cependant, il n'est pas facile de créer des règles spécifiques afin qu'un service particulier ne puisse communiquer qu'avec un autre service spécifique. Les services ne disposent d'aucun port spécifique à utiliser pour les règles de mise en réseau des groupes de sécurité.

Options de mise en réseau des tâches Amazon ECS pour le type de lancement Fargate

Par défaut, chaque tâche Amazon ECS sur Fargate reçoit une interface réseau Elastic (ENI) avec une adresse IP privée principale. Lorsque vous utilisez un sous-réseau public, vous pouvez éventuellement attribuer une adresse IP publique à l'ENI de la tâche. Si votre VPC est configuré pour le mode double pile et que vous utilisez un sous-réseau avec un bloc IPv6 CIDR, l'ENI de votre tâche reçoit également une adresse. IPv6 Une tâche ne peut avoir qu'une seule ENI associée à la fois. Les conteneurs qui appartiennent à la même tâche peuvent également communiquer via l'interface `localhost`. Pour plus d'informations sur les sous-réseaux VPCs et les sous-réseaux, consultez Comment fonctionne Amazon VPC dans le guide de l'utilisateur Amazon VPC.

Pour qu'une tâche sur Fargate puisse extraire une image de conteneur, la tâche doit avoir un routage vers Internet. L'exemple suivant décrit également comment vous pouvez vérifier le routage vers Internet pour votre tâche.

- Lorsque vous utilisez un sous-réseau public, vous pouvez attribuer une adresse IP publique à l'ENI de la tâche.
- Lorsque vous utilisez un sous-réseau privé, une passerelle NAT peut être attachée au sous-réseau.
- Lorsque vous utilisez des images de conteneur hébergées dans Amazon ECR, vous pouvez configurer Amazon ECR pour utiliser un point de terminaison VPC d'interface et l'extraction de l'image se fait via l'adresse privée de la tâche. IPv4 Pour plus d'informations, consultez Points de terminaison d’un VPC d'interface Amazon ECR (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.

Dans la mesure où chaque tâche obtient sa propre ENI, vous pouvez utiliser des fonctionnalités de mise en réseau telles que les journaux de flux VPC, que vous pouvez utiliser pour surveiller le trafic vers et depuis vos tâches. Pour plus d'informations, consultez la rubrique Journaux de flux VPC dans le Guide de l'utilisateur Amazon VPC.

Vous pouvez également en profiter AWS PrivateLink. Vous pouvez configurer un point de terminaison d'interface VPC afin de pouvoir accéder à Amazon ECS APIs via des adresses IP privées. AWS PrivateLink restreint tout le trafic réseau entre votre VPC et Amazon ECS vers le réseau Amazon. Vous n'avez pas besoin d'une passerelle Internet, d'un périphérique NAT ni d'une passerelle privée virtuelle. Pour plus d'informations, consultez la section Points de terminaison VPC de l'interface Amazon ECS ().AWS PrivateLink

Pour des exemples d'utilisation de la `NetworkConfiguration` ressource avec AWS CloudFormation, voirthe section called "Exemples de AWS CloudFormation modèles".

Les ENIs qui sont créés sont entièrement gérés par AWS Fargate. De plus, une politique IAM associée est utilisée pour accorder des autorisations pour Fargate. Pour les tâches utilisant la version `1.4.0` de la plateforme Fargate ou une version ultérieure, la tâche reçoit une seule ENI (appelée ENI de tâche) et tout le trafic réseau passe par cette ENI au sein de votre VPC. Ce trafic est enregistré dans les journaux de flux de votre VPC. Pour les tâches qui utilisent la version `1.3.0` de la plateforme Fargate et les versions antérieures, en plus de l'ENI de la tâche, la tâche reçoit également une ENI distincte détenue par Fargate, qui est utilisée pour certains trafics réseau qui ne sont pas visibles dans les journaux de flux VPC. Le tableau suivant décrit le comportement du trafic réseau et la stratégie IAM requise pour chaque version de plateforme.

Action	Flux de trafic avec les versions **`1.3.0`** et antérieures de la plateforme Linux	Flux de trafic avec la version **`1.4.0`** de la plateforme Linux	Flux de trafic avec la version **`1.0.0`** de la plateforme Windows	Autorisation IAM
Récupération des informations d'identification de connexion Amazon ECR	ENI Fargate	ENI de tâche	ENI de tâche	Rôle IAM d'exécution de tâche

Action	Flux de trafic avec les versions **1.3.0** et antérieures de la plateforme Linux	Flux de trafic avec la version **1.4.0** de la plateforme Linux	Flux de trafic avec la version **1.0.0** de la plateforme Windows	Autorisation IAM
Extraction d'image	ENI de tâche	ENI de tâche	ENI de tâche	Rôle IAM d'exécution de tâche
Envoi de journaux via un pilote de journal	ENI de tâche	ENI de tâche	ENI de tâche	Rôle IAM d'exécution de tâche
Envoi de journaux FireLens pour Amazon ECS	ENI de tâche	ENI de tâche	ENI de tâche	Rôle IAM de tâche
Récupération de secrets à partir de Secrets Manager ou de Systems Manager	ENI Fargate	ENI de tâche	ENI de tâche	Rôle IAM d'exécution de tâche
Trafic du système de fichiers Amazon EFS	Non disponible	ENI de tâche	ENI de tâche	Rôle IAM de tâche
Trafic d'application	ENI de tâche	ENI de tâche	ENI de tâche	Rôle IAM de tâche

Considérations

Tenez compte des éléments suivants lorsque vous utilisez le réseau de tâches.

- Le rôle lié au service Amazon ECS est requis pour fournir à Amazon ECS les autorisations nécessaires pour passer des appels vers d'autres AWS services en votre nom. Ce rôle est généré lorsque vous créez un cluster, ou si vous créez ou mettez à jour un service dans la AWS Management Console. Pour de plus amples informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS. Vous pouvez également créer le rôle lié à un service à l'aide de la commande suivante AWS CLI .

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- Amazon ECS remplit le nom d'hôte de la tâche avec un nom d'hôte DNS fourni par Amazon lorsque les options `enableDnsHostnames` et `enableDnsSupport` sont activées sur votre VPC. Si ces options ne sont pas activées, le nom d'hôte DNS de la tâche est un nom d'hôte aléatoire. Pour plus d'informations sur les paramètres DNS d'un VPC, veuillez consulter la rubrique Utilisation de DNS avec votre VPC dans le Guide de l'utilisateur Amazon VPC.
- Vous pouvez uniquement spécifier jusqu'à 16 sous-réseaux et 5 groupes de sécurité pour `awsVpcConfiguration`. Pour plus d'informations, consultez le AwsVpcConfigurationmanuel Amazon Elastic Container Service API Reference.
- Vous ne pouvez pas détacher ou modifier manuellement ceux ENIs créés et attachés par Fargate. L'objectif est d'empêcher la suppression accidentelle de toute ENI associée à une tâche en cours d'exécution. Pour libérer le ENIs pour une tâche, arrêtez la tâche.
- Si un sous-réseau VPC est mis à jour pour modifier l'ensemble d'options DHCP qu'il utilise, vous ne pouvez pas également appliquer ces modifications aux tâches existantes qui utilisent le VPC. Démarrez de nouvelles tâches, qui recevront le nouveau paramètre pour migrer en douceur tout en testant la nouvelle modification, puis arrêtez les anciennes, si aucune restauration n'est requise.
- Ce qui suit s'applique aux tâches exécutées sur la `1.4.0` version de la plateforme Fargate ou ultérieure pour Linux ou pour Windows. `1.0.0` Les tâches lancées dans des sous-réseaux à double pile reçoivent une IPv4 adresse et une IPv6 adresse.
- Pour les tâches qui utilisent une version de plate-forme `1.4.0` ou ultérieure pour Linux ou `1.0.0` Windows, la tâche ENIs prend en charge les cadres jumbo. Les interfaces réseau sont configurées avec une unité de transmission maximale (MTU), qui correspond à la taille de la charge utile la plus élevée possible dans une seule trame. Plus la MTU est importante, plus la charge utile de l'application peut s'intégrer dans une seule trame. Cela réduit les frais généraux par trame et augmente l'efficacité. La prise en charge des trames jumbo limite la surcharge lorsque le chemin réseau entre votre tâche et la destination prend en charge les trames jumbo.
- Les services comportant des tâches qui utilisent le type de lancement Fargate prennent uniquement en charge les équilibreurs de charge Application Load Balancer et Network Load

Balancer. L'équilibreur de charge Classic Load Balancer n'est pas pris en charge. Lorsque vous créez des groupes cibles pour ces services, vous devez choisir le type de cible `ip`, et non `instance`. Pour de plus amples informations, veuillez consulter Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS.

Utilisation d'un VPC en mode double pile

Lorsque vous utilisez un VPC en mode double pile, vos tâches peuvent communiquer entre elles IPv4 ou IPv6 par les deux. IPv4 et les IPv6 adresses sont indépendantes les unes des autres et vous devez configurer le routage et la sécurité dans votre VPC séparément pour IPv4 et. IPv6 Pour plus d'informations sur la configuration de votre VPC pour le mode double pile, consultez la section Migration vers le guide de l'utilisateur IPv6 Amazon VPC.

Si les conditions suivantes sont remplies, une adresse est attribuée aux tâches Amazon ECS sur Fargate : IPv6

- Les paramètres de votre `dualStackIPv6` compte Amazon ECS sont activés (`enabled`) pour que le principal IAM lance vos tâches dans la région dans laquelle vous les lancez. Ce paramètre ne peut être modifié qu'à l'aide de l'API ou AWS CLI. Vous avez la possibilité d'activer ce paramètre pour un principal IAM spécifique sur votre compte ou pour l'ensemble de votre compte en définissant les paramètres par défaut de votre compte. Pour de plus amples informations, veuillez consulter Accédez aux fonctionnalités d'Amazon ECS avec les paramètres du compte.
- Votre VPC et votre sous-réseau sont activés pour. IPv6 Pour plus d'informations sur la configuration de votre VPC pour le mode double pile, consultez la section Migration vers le guide de l'utilisateur IPv6 Amazon VPC.
- Votre sous-réseau est activé pour l'attribution automatique d'adresses IPv6 . Pour plus d'informations sur la configuration de votre sous-réseau, consultez Modifier l'attribut d' IPv6 adressage de votre sous-réseau dans le guide de l'utilisateur Amazon VPC.
- La tâche ou le service utilise la version de plateforme `1.4.0` Fargate ou une version ultérieure pour Linux.

Si vous configurez votre VPC avec une passerelle Internet ou une passerelle Internet sortante uniquement, les tâches Amazon ECS sur Fargate auxquelles une adresse a été attribuée peuvent accéder à Internet. IPv6 Les passerelles NAT ne sont pas nécessaires. Pour plus d'informations, veuillez consulter les rubriques Passerelles Internet et Passerelle Internet en sortie uniquement dans le Guide de l'utilisateur Amazon VPC.

Options de stockage pour les tâches Amazon ECS

Amazon ECS vous propose des options de stockage de easy-to-use données flexibles et économiques en fonction de vos besoins. Amazon ECS prend en charge les options de volume de données suivantes pour les conteneurs :

Volume de données	Types de lancement pris en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
Amazon Elastic Block Store (Amazon EBS)	Fargate, Amazon EC2	Linux	Peut être conservé lorsqu'il est associé à une tâche autonome. Éphémère lorsqu'il est rattaché à une tâche gérée par un service.	Les volumes Amazon EBS fournissent un stockage par blocs rentable, durable et performant pour les charges de travail conteneur isées gourmandes en données. Les cas d'utilisation courants incluent les charges de travail transactionnelles telles que les bases de données, les bureaux virtuels et les volumes racines, et les charges de travail gourmandes en débit telles que le traitemen

Volume de données	Types de lancement pris en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				t des journaux et les charges de travail ETL. Pour de plus amples informations, veuillez consulter Utiliser des volumes Amazon EBS avec Amazon ECS.

Volume de données	Types de lancement pris en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
Amazon Elastic File System (Amazon EFS)	Fargate, Amazon EC2	Linux	Persistante	Les volumes Amazon EFS fournissent un stockage de fichiers partagé simple, évolutif et permanent à utiliser avec vos tâches Amazon ECS, qui augmente et diminue automatiq uement au fur et à mesure que vous ajoutez et supprimez des fichiers. Les volumes Amazon EFS prennent en charge la simultanéité et sont utiles pour les applications conteneurisées qui évoluent horizontalement et nécessitent des fonctionn alités de stockage telles qu'une faible

Volume de données	Types de lancement pris en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				latence, un débit élevé et une cohérence. read-after-write Les cas d'utilisation courants incluent les charges de travail telles que l'analyse des données, le traitemen t multimédia, la gestion de contenu et la diffusion Web. Pour de plus amples informati ons, veuillez consulter Utiliser les volumes Amazon EFS avec Amazon ECS.

Volume de données	Types de lancement pris en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
Serveur FSx de fichiers Amazon pour Windows	Amazon EC2	Windows	Persistante	FSx pour Windows File Server, les volumes fournissent des serveurs de fichiers Windows entièreme nt gérés que vous pouvez utiliser pour configurer vos tâches Windows nécessitant un stockage de fichiers persistan t, distribué , partagé et statique. Les cas d'utilisation courants incluent les applicati ons .NET qui peuvent nécessiter des dossiers locaux comme stockage persistant pour enregistrer les sorties des applications. Le serveur

Volume de données	Types de lancement pris en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				de fichiers Amazon FSx pour Windows propose un dossier local dans le conteneur qui permet à plusieurs conteneurs de lire et écrire sur le même système de fichiers soutenu par un partage SMB. Pour de plus amples informations, veuillez consulter Utilisation FSx pour les volumes de serveurs de fichiers Windows avec Amazon ECS.

Volume de données	Types de lancement pris en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
Volumes Docker	Amazon EC2	Windows, Linux	Persistante	Les volumes Docker sont une fonctionnalité du moteur d'exécution des conteneurs Docker qui permet aux conteneurs de conserver les données en montant un répertoire à partir du système de fichiers de l'hôte. Les pilotes de volume Docker (également appelés plugins) sont utilisés pour intégrer des volumes de conteneurs à des systèmes de stockage externes. Les volumes Docker peuvent être gérés par des pilotes tiers ou par le `local` pilote intégré.

Volume de données	Types de lancement pris en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				Les cas d'utilisation courants des volumes Docker incluent la fourniture de volumes de données persistants ou le partage de volumes à différents emplacements sur différents conteneurs d'une même instance de conteneur. Pour de plus amples informations, veuillez consulter Utiliser des volumes Docker avec Amazon ECS.

Volume de données	Types de lancement pris en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
Montages liés	Fargate, Amazon EC2	Windows, Linux	Éphémère	Les montages par liaison consistent en un fichier ou un répertoire sur l'hôte, tel qu'une EC2 instance Amazon ou AWS Fargate un répertoire monté sur un conteneur . Les cas d'utilisation courants des montages par liaison incluent le partage d'un volume d'un conteneur source avec d'autres conteneurs dans le cadre de la même tâche, ou le montage d'un volume hôte ou d'un volume vide dans un ou plusieurs conteneurs. Pour de plus amples informations, veuillez

Volume de données	Types de lancement pris en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				consulter Utiliser des montages par liaison avec Amazon ECS.

Utiliser des volumes Amazon EBS avec Amazon ECS

Les volumes Amazon Elastic Block Store (Amazon EBS) fournissent un stockage par blocs hautement disponible, rentable, durable et performant pour les charges de travail gourmandes en données. Les volumes Amazon EBS peuvent être utilisés avec les tâches Amazon ECS pour les applications à haut débit et gourmandes en transactions.

Lors du lancement d'une tâche autonome, vous pouvez fournir la configuration qui sera utilisée pour associer un volume EBS à la tâche. Lors de la création ou de la mise à jour du service, vous pouvez fournir la configuration qui sera utilisée pour associer un volume EBS par tâche à chaque tâche gérée par le service Amazon ECS.

En fournissant la configuration des volumes au moment du lancement plutôt que dans la définition des tâches, vous créez des définitions de tâches qui ne sont pas limitées à un type de volume de données ou à des paramètres de volume EBS spécifiques. Vous pouvez ensuite réutiliser vos définitions de tâches dans différents environnements d'exécution. Par exemple, vous pouvez fournir un débit supérieur lors du déploiement de vos charges de travail de production par rapport à vos environnements de pré-production.

Les volumes Amazon EBS attachés aux tâches Amazon ECS sont gérés par Amazon ECS en votre nom. Les volumes peuvent être chiffrés avec des clés AWS Key Management Service (AWS KMS) pour protéger vos données. Vous pouvez soit configurer de nouveaux volumes vides pour les pièces jointes, soit utiliser des instantanés pour charger des données à partir de volumes existants. Lorsque vous utilisez des instantanés pour configurer des volumes, vous pouvez spécifier`volumeInitializationRate`, en Mbits/s, le niveau auquel les données sont extraites du cliché afin de créer des volumes entièrement initialisés dans un délai prévisible. Pour plus d'informations sur l'initialisation des volumes, consultez Initialiser les volumes Amazon EBS dans le guide de l'utilisateur Amazon EBS. Pour plus d'informations sur la configuration des volumes Amazon EBS, consultez Différer la configuration du volume à l'heure de lancement dans une définition

de tâche Amazon ECS etSpécifiez la configuration du volume Amazon EBS lors du déploiement d'Amazon ECS.

Pour surveiller les performances de votre volume, vous pouvez également utiliser CloudWatch les métriques Amazon. Pour plus d'informations sur les métriques Amazon ECS pour les volumes Amazon EBS, consultez CloudWatch Métriques Amazon ECS et les métriques Amazon ECS Container Insights.

L'association d'un volume Amazon EBS à une tâche est prise en charge dans toutes les entreprises et en Chine Régions AWSqui prennent en charge Amazon ECS.

Pour plus d'informations sur les volumes Amazon EBS, consultez la section consacrée aux volumes Amazon EBS dans le guide de l'utilisateur Amazon EBS.

Systèmes d'exploitation et types de lancement pris en charge

Le tableau suivant indique les configurations de système d'exploitation et de type de lancement prises en charge.

Type de lancement	Linux	Windows
Fargate	Les volumes Amazon EBS sont pris en charge sur la version de plate-forme `1.4.0` ou ultérieure (Linux). Pour de plus amples informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.	Non pris en charge
EC2	Tâches hébergées sur des instances Linux Nitro basées sur Amazon Machine Images () AMIs optimisées pour Amazon ECS. Pour plus d'informations sur les types d'instances, consultez la section Types d'instances	Tâches hébergées sur des instances Linux Nitro basées sur Amazon Machine Images () AMIs optimisées pour Amazon ECS. Pour plus d'informations sur les types d'instances, consultez la section Types d'instances

Type de lancement	Linux	Windows
	dans le guide de EC2 l'utilisa teur Amazon. Les volumes Amazon EBS sont pris en charge sur une `20231219` AMI optimisée pour ECS ou une version ultérieure. Pour plus d'informa tions, consultez Extraction des métadonnées d'AMI optimisée pour Amazon ECS.	dans le guide de EC2 l'utilisa teur Amazon. Les volumes Amazon EBS sont pris en charge sur une `20241017` AMI optimisée pour ECS ou une version ultérieur e. Pour plus d'informations, consultez la section Extractio n des métadonnées d'AMI Windows optimisées pour Amazon ECS.

Considérations

Tenez compte des points suivants lorsque vous utilisez des volumes Amazon EBS :

- Vous ne pouvez pas configurer de volumes Amazon EBS pour les associer à des tâches `use1-az3` Amazon ECS de type Fargate Launch dans la zone de disponibilité.
- Le type de volume magnétique (`standard`) Amazon EBS n'est pas pris en charge pour les tâches hébergées sur Fargate. Pour plus d'informations sur les types de volumes Amazon EBS, consultez les volumes Amazon EBS dans le guide de EC2 l'utilisateur Amazon.
- Un rôle IAM dans l'infrastructure Amazon ECS est requis lors de la création d'un service ou d'une tâche autonome qui consiste à configurer un volume lors du déploiement. Vous pouvez associer la stratégie `AmazonECSInfrastructureRolePolicyForVolumes` IAM AWS gérée au rôle, ou vous pouvez utiliser la stratégie gérée comme guide pour créer et associer votre propre politique avec des autorisations répondant à vos besoins spécifiques. Pour de plus amples informations, veuillez consulter Rôle IAM dans l'infrastructure Amazon ECS.
- Vous pouvez associer au maximum un volume Amazon EBS à chaque tâche Amazon ECS, et il doit s'agir d'un nouveau volume. Vous ne pouvez pas associer un volume Amazon EBS existant à une tâche. Toutefois, vous pouvez configurer un nouveau volume Amazon EBS lors du déploiement à l'aide de l'instantané d'un volume existant.

- Pour utiliser les volumes Amazon EBS avec les services Amazon ECS, le contrôleur de déploiement doit être « ECS ». Les stratégies de déploiement et blue/green de déploiement sont prises en charge lors de l'utilisation de ce contrôleur de déploiement.
- Pour qu'un conteneur de votre tâche puisse écrire sur le volume Amazon EBS monté, vous devez exécuter le conteneur en tant qu'utilisateur root.
- Amazon ECS ajoute automatiquement les balises réservées `AmazonECSCreated` et les ajoute `AmazonECSManaged` au volume attaché. Si vous supprimez ces balises du volume, Amazon ECS ne sera pas en mesure de gérer le volume en votre nom. Pour plus d'informations sur le balisage des volumes Amazon EBS, consultez la section Marquage des volumes Amazon EBS. Pour plus d'informations sur le balisage des ressources Amazon ECS, consultez la section Marquage de vos ressources Amazon ECS.
- Le provisionnement de volumes à partir d'un instantané d'un volume Amazon EBS contenant des partitions n'est pas pris en charge.
- Les volumes attachés à des tâches gérées par un service ne sont pas conservés et sont toujours supprimés à la fin de la tâche.
- Vous ne pouvez pas configurer de volumes Amazon EBS pour les associer à des tâches Amazon ECS en cours d'exécution. AWS Outposts

Différer la configuration du volume à l'heure de lancement dans une définition de tâche Amazon ECS

Pour configurer un volume Amazon EBS à associer à votre tâche, vous devez spécifier la configuration du point de montage dans votre définition de tâche et nommer le volume. Vous devez également `configuredAtLaunch` définir cette valeur, `true` car les volumes Amazon EBS ne peuvent pas être configurés pour être joints dans la définition de tâche. Au lieu de cela, les volumes Amazon EBS sont configurés pour être joints lors du déploiement.

Pour enregistrer la définition de tâche à l'aide de AWS Command Line Interface (AWS CLI), enregistrez le modèle sous forme de fichier JSON, puis transmettez-le comme entrée pour la `register-task-definition` commande.

Pour créer et enregistrer une définition de tâche à l'aide du AWS Management Console, voirCréation d'une définition de tâche Amazon ECS à l'aide de la console.

La définition de tâche suivante indique la syntaxe des `volumes` objets `mountPoints` et contenus dans la définition de tâche. Pour plus d'informations sur les paramètres de définition des tâches, consultezParamètres de définition des tâches Amazon ECS pour le type de lancement Fargate. Pour utiliser cet exemple, remplacez *`user input placeholders`* par vos propres informations.

Linux

```
{
    "family": "mytaskdef",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/nginx/nginx:latest",
            "networkMode": "awsvpc",
           "portMappings": [
                {
                    "name": "nginx-80-tcp",
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp",
                    "appProtocol": "http"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEBSVolume",
                    "containerPath": "/mount/ebs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEBSVolume",
            "configuredAtLaunch": true
        }
    ],
    "requiresCompatibilities": [
        "FARGATE", "EC2"
    ],
    "cpu": "1024",
    "memory": "3072",
    "networkMode": "awsvpc"
}
```

Windows

```
{
    "family": "mytaskdef",
     "memory": "4096",
     "cpu": "2048",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["EC2"]
    "containerDefinitions": [
        {
             "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 443,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEBSVolume",
                    "containerPath": "drive:\ebs",
                    "readOnly": true
                }
            ]
        }
    ],
```

```
    "volumes": [
        {
            "name": "myEBSVolume",
            "configuredAtLaunch": true
        }
    ],
    "requiresCompatibilities": [
        "FARGATE", "EC2"
    ],
    "cpu": "1024",
    "memory": "3072",
    "networkMode": "awsvpc"
}
```

`mountPoints`

Type : tableau d'objets

Obligatoire : non

Les points de montage des volumes de données de votre conteneur. Ce paramètre correspond `Volumes` à l'API Docker create-container et `--volume` à l'option docker run.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`. Les conteneurs Windows ne peuvent pas monter de répertoires sur un autre lecteur, et les points de montage ne peuvent pas être utilisés sur plusieurs lecteurs. Vous devez spécifier des points de montage pour associer un volume Amazon EBS directement à une tâche Amazon ECS.

`sourceVolume`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Le chemin dans le conteneur où le volume sera monté.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

Pour les tâches exécutées sur EC2 des instances exécutant le système d'exploitation Windows, laissez la valeur par défaut de`false`.

`name`

Type : chaîne

Obligatoire : non

Nom du volume. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, tirets () et traits de soulignement (`-`) sont autorisés. _ Ce nom est référencé dans le `sourceVolume` paramètre de l'`mountPoints`objet de définition du conteneur.

`configuredAtLaunch`

Type : booléen

Obligatoire : Oui, lorsque vous souhaitez associer un volume EBS directement à une tâche.

Spécifie si un volume est configurable au lancement. Lorsque ce paramètre est défini sur`true`, vous pouvez configurer le volume lorsque vous exécutez une tâche autonome, ou lorsque vous créez ou mettez à jour un service. Lorsque ce paramètre est défini sur`false`, vous ne pourrez pas fournir d'autre configuration de volume dans la définition de tâche. Ce paramètre doit être fourni et défini sur `true` pour configurer un volume Amazon EBS à associer à une tâche.

Chiffrez les données stockées dans les volumes Amazon EBS associés aux tâches Amazon ECS

Vous pouvez utiliser AWS Key Management Service (AWS KMS) pour créer et gérer des clés cryptographiques qui protègent vos données. Les volumes Amazon EBS sont chiffrés au repos à l'aide AWS KMS keys de. Les types de données suivants sont cryptés :

- Données stockées au repos sur le volume
- E/S de disque

- Instantanés créés à partir du volume
- Nouveaux volumes créés à partir de snapshots chiffrés

Les volumes Amazon EBS attachés à des tâches peuvent être chiffrés en utilisant soit une valeur par défaut Clé gérée par AWS avec l'alias`alias/aws/ebs`, soit une clé symétrique gérée par le client spécifiée dans la configuration du volume. Clés gérées par AWS Les valeurs par défaut sont uniques à chaque Compte AWS Région AWS utilisateur et sont créées automatiquement. Pour créer une clé symétrique gérée par le client, suivez les étapes décrites dans la section Création de clés KMS de chiffrement symétriques dans le manuel du AWS KMS développeur.

Vous pouvez configurer le chiffrement Amazon EBS par défaut afin que tous les nouveaux volumes créés et attachés à une tâche spécifique Région AWS soient chiffrés à l'aide de la clé KMS que vous spécifiez pour votre compte. Pour plus d'informations sur le chiffrement Amazon EBS et le chiffrement par défaut, consultez la section Chiffrement Amazon EBS dans le guide de l'utilisateur Amazon EBS.

Vous pouvez également configurer le chiffrement au niveau du cluster Amazon ECS pour le stockage géré par Amazon ECS lorsque vous créez ou mettez à jour un cluster. Le chiffrement au niveau du cluster peut être utilisé pour chiffrer tous les volumes Amazon EBS associés à des tâches exécutées dans un cluster spécifique à l'aide de la clé KMS spécifiée au niveau du cluster. Pour plus d'informations sur la configuration du chiffrement au niveau du cluster, consultez ManagedStorageConfigurationla référence de l'API Amazon ECS.

Vous pouvez configurer n'importe quelle combinaison de ces touches. L'ordre de priorité des clés KMS est le suivant :

1. La clé KMS spécifiée dans la configuration du volume. Lorsque vous spécifiez une clé KMS dans la configuration du volume, elle remplace la clé par défaut d'Amazon EBS et toute clé KMS spécifiée au niveau du cluster.
2. La clé KMS spécifiée au niveau du cluster. Lorsque vous spécifiez une clé KMS pour le chiffrement au niveau du cluster du stockage géré Amazon ECS, elle remplace le chiffrement par défaut d'Amazon EBS mais ne remplace aucune clé KMS spécifiée dans la configuration du volume.
3. Chiffrement par défaut d'Amazon EBS. Le chiffrement par défaut s'applique lorsque vous ne spécifiez ni clé KMS au niveau du cluster ni clé dans la configuration du volume. Si vous activez le chiffrement Amazon EBS par défaut, il s'agit de la clé KMS que vous spécifiez pour le chiffrement par défaut. Dans le cas contraire, la valeur par défaut est le Clé gérée par AWS avec l'alias`alias/aws/ebs`.

> **Note**
>
> Si vous définissez cette `encrypted` option `false` dans la configuration de votre volume, que vous ne spécifiez aucune clé KMS au niveau du cluster et que vous activez le chiffrement Amazon EBS par défaut, le volume sera toujours chiffré avec la clé spécifiée pour le chiffrement Amazon EBS par défaut.

Politique relative aux clés KMS gérée par le client

Pour chiffrer un volume EBS attaché à votre tâche à l'aide d'une clé gérée par le client, vous devez configurer votre politique de clé KMS afin de vous assurer que le rôle IAM que vous utilisez pour la configuration du volume dispose des autorisations nécessaires pour utiliser la clé. La politique clé doit inclure les `kms:GenerateDataKey*` autorisations `kms:CreateGrant` et. Les `kms:ReEncryptFrom` autorisations `kms:ReEncryptTo` et sont nécessaires pour chiffrer les volumes créés à l'aide de snapshots. Si vous souhaitez configurer et chiffrer uniquement les nouveaux volumes vides à des fins de pièce jointe, vous pouvez exclure les `kms:ReEncryptFrom` autorisations `kms:ReEncryptTo` et.

L'extrait de code JSON suivant montre les principales déclarations de politique que vous pouvez associer à votre politique clé KMS. L'utilisation de ces instructions permettra à Amazon ECS d'utiliser la clé pour chiffrer le volume EBS. Pour utiliser les exemples de déclarations de politique, *`user input placeholders`* remplacez-les par vos propres informations. Comme toujours, configurez uniquement les autorisations dont vous avez besoin.

```
{
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:DescribeKey",
      "Resource":"*"
    },
    {
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": [
      "kms:GenerateDataKey*",
      "kms:ReEncryptTo",
      "kms:ReEncryptFrom"
      ],
```

```
      "Resource":"*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "aws_account_id",
          "kms:ViaService": "ec2.region.amazonaws.com"
        },
        "ForAnyValue:StringEquals": {
          "kms:EncryptionContextKeys": "aws:ebs:id"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:CreateGrant",
      "Resource":"*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "aws_account_id",
          "kms:ViaService": "ec2.region.amazonaws.com"
        },
        "ForAnyValue:StringEquals": {
          "kms:EncryptionContextKeys": "aws:ebs:id"
        },
        "Bool": {
          "kms:GrantIsForAWSResource": true
        }
      }
    }
```

Pour plus d'informations sur les politiques et autorisations clés, voir Politiques clés AWS KMS et AWS KMS autorisations dans le Guide du AWS KMS développeur. Pour résoudre les problèmes de connexion aux volumes EBS liés aux autorisations clés, consultezRésolution des problèmes liés aux pièces jointes de volumes Amazon EBS aux tâches Amazon ECS .

Spécifiez la configuration du volume Amazon EBS lors du déploiement d'Amazon ECS

Après avoir enregistré une définition de tâche dont le `configuredAtLaunch` paramètre est défini sur`true`, vous pouvez configurer un volume Amazon EBS lors du déploiement, lorsque vous exécutez une tâche autonome, ou lorsque vous créez ou mettez à jour un service.

Pour configurer un volume, vous pouvez utiliser Amazon ECS APIs ou transmettre un fichier JSON en entrée pour les AWS CLI commandes suivantes :

- `run-task`pour exécuter une tâche ECS autonome.
- `start-task`pour exécuter une tâche ECS autonome dans une instance de conteneur spécifique. Cette commande ne s'applique pas aux tâches de type lancement Fargate.
- `create-service`pour créer un nouveau service ECS.
- `update-service`pour mettre à jour un service existant.

> **Note**
>
> Pour qu'un conteneur de votre tâche puisse écrire sur le volume Amazon EBS monté, vous devez exécuter le conteneur en tant qu'utilisateur root.

Vous pouvez également configurer un volume Amazon EBS à l'aide du AWS Management Console. Pour plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS, Création d'un déploiement de mises à jour continues Amazon ECS et Mettre à jour un service Amazon ECS.

L'extrait de code JSON suivant montre tous les paramètres d'un volume Amazon EBS qui peuvent être configurés lors du déploiement. Pour utiliser ces paramètres pour la configuration des volumes, *`user input placeholders`* remplacez-les par vos propres informations. Pour plus d'informations sur ces paramètres, consultez la section Configurations des volumes.

```
"volumeConfigurations": [
        {
            "name": "ebs-volume",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
                "volumeType": "gp3",
                "sizeInGiB": 10,
                "snapshotId": "snap-12345",
                "volumeInitializationRate":100,
                "iops": 3000,
                "throughput": 125,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
```

```
                            {
                                "key": "key1",
                                "value": "value1"
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "arn:aws:iam::1111222333:role/ecsInfrastructureRole",
                 "terminationPolicy": {
                    "deleteOnTermination": true//can't be configured for service-
managed tasks, always true
                },
                "filesystemType": "ext4"
            }
        }
    ]
```

> **Important**
>
> Assurez-vous que ce que `volumeName` vous spécifiez dans la configuration est identique à celui que `volumeName` vous spécifiez dans votre définition de tâche.

Pour plus d'informations sur la vérification de l'état de la connexion au volume, consultezRésolution des problèmes liés aux pièces jointes de volumes Amazon EBS aux tâches Amazon ECS . Pour plus d'informations sur le rôle d'infrastructure Amazon ECS AWS Identity and Access Management (IAM) nécessaire à l'attachement d'un volume EBS, consultez. Rôle IAM dans l'infrastructure Amazon ECS

Voici des exemples d'extraits de code JSON illustrant la configuration des volumes Amazon EBS. Ces exemples peuvent être utilisés en enregistrant les extraits dans des fichiers JSON et en transmettant les fichiers en tant que paramètres (en utilisant le `--cli-input-json file://filename` paramètre) pour AWS CLI les commandes. Remplacez *`user input placeholders`* par vos propres informations.

Configuration d'un volume pour une tâche autonome

L'extrait suivant montre la syntaxe permettant de configurer les volumes Amazon EBS pour les associer à une tâche autonome. L'extrait de code JSON suivant montre la syntaxe de configuration des paramètres`volumeType`, `sizeInGiBencrypted`, et`kmsKeyId`. La configuration spécifiée dans le fichier JSON est utilisée pour créer et associer un volume EBS à la tâche autonome.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "volumeConfigurations": [
        {
            "name": "datadir",
            "managedEBSVolume": {
                "volumeType": "gp3",
                "sizeInGiB": 100,
                "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
                "encrypted": true,
                "kmsKeyId":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        }
   ]
}
```

Configuration d'un volume lors de la création du service

L'extrait suivant montre la syntaxe permettant de configurer les volumes Amazon EBS pour les associer à des tâches gérées par un service. Les volumes proviennent de l'instantané spécifié à l'aide du `snapshotId` paramètre à un débit de 200 Mbits/s. La configuration spécifiée dans le fichier JSON est utilisée pour créer et associer un volume EBS à chaque tâche gérée par le service.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
              "snapshotId": "snap-12345",
              "volumeInitializationRate": 200
            }
        }
   ]
}
```

Configuration d'un volume lors de la mise à jour du service

L'extrait de code JSON suivant montre la syntaxe de mise à jour d'un service pour lequel aucun volume Amazon EBS n'était auparavant configuré pour être rattaché à des tâches. Vous devez fournir l'ARN d'une révision de définition de tâche `configuredAtLaunch` définie sur`true`. L'extrait de code JSON suivant montre la syntaxe permettant de configurer les paramètres `volumeType` `sizeInGiBthroughput`,`iops`, et`filesystemType`. Cette configuration est utilisée pour créer et associer un volume EBS à chaque tâche gérée par le service.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "service": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
               "volumeType": "gp3",
                "sizeInGiB": 100,
                 "iops": 3000,
                "throughput": 125,
                "filesystemType": "ext4"
            }
        }
   ]
}
```

Configurer un service pour ne plus utiliser les volumes Amazon EBS

L'extrait de code JSON suivant montre la syntaxe permettant de mettre à jour un service afin qu'il n'utilise plus les volumes Amazon EBS. Vous devez fournir l'ARN d'une définition de tâche `configuredAtLaunch` définie sur ou `false` d'une définition de tâche sans le `configuredAtLaunch` paramètre. Vous devez également fournir un `volumeConfigurations` objet vide.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "service": "mysvc",
```

```
    "desiredCount": 2,
    "volumeConfigurations": []
}
```

Politique de résiliation pour les volumes Amazon EBS

Lorsqu'une tâche Amazon ECS prend fin, Amazon ECS utilise la `deleteOnTermination` valeur pour déterminer si le volume Amazon EBS associé à la tâche terminée doit être supprimé. Par défaut, les volumes EBS attachés à des tâches sont supprimés lorsque la tâche est arrêtée. Pour les tâches autonomes, vous pouvez modifier ce paramètre afin de préserver le volume à la fin de la tâche.

> **Note**
>
> Les volumes attachés à des tâches gérées par un service ne sont pas conservés et sont toujours supprimés à la fin de la tâche.

Étiqueter les volumes Amazon EBS

Vous pouvez baliser les volumes Amazon EBS à l'aide de l'`tagSpecifications`objet. À l'aide de l'objet, vous pouvez fournir vos propres balises et définir la propagation des balises à partir de la définition de la tâche ou du service, selon que le volume est attaché à une tâche autonome ou à une tâche d'un service. Le nombre maximum de balises pouvant être attachées à un volume est de 50.

> **Important**
>
> Amazon ECS attache automatiquement les balises `AmazonECSCreated` et les balises `AmazonECSManaged` réservées à un volume Amazon EBS. Cela signifie que vous pouvez contrôler l'attachement d'un maximum de 48 balises supplémentaires à un volume. Ces balises supplémentaires peuvent être définies par l'utilisateur, gérées par ECS ou propagées.

Si vous souhaitez ajouter des balises gérées par Amazon ECS à votre volume, vous devez les `enableECSManagedTags` définir `true` dans votre `UpdateService``CreateService`, `RunTask` ou `StartTask` appeler. Si vous activez les balises gérées par Amazon ECS, Amazon ECS balisera automatiquement le volume avec des informations sur le cluster et le service (`aws:ecs:`*`clusterName`*etaws`:ecs:`*`serviceName`*). Pour plus d'informations sur le balisage des ressources Amazon ECS, consultez la section Marquage de vos ressources Amazon ECS.

L'extrait de code JSON suivant montre la syntaxe permettant de baliser chaque volume Amazon EBS associé à chaque tâche d'un service avec une balise définie par l'utilisateur. Pour utiliser cet exemple pour créer un service, remplacez-le *user input placeholders* par vos propres informations.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "enableECSManagedTags": true,
   "volumeConfigurations": [
        {
            "name": "datadir",
            "managedEBSVolume": {
                "volumeType": "gp3",
                "sizeInGiB": 100,
                 "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "key1",
                                "value": "value1"
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn":"arn:aws:iam:1111222333:role/ecsInfrastructureRole",
                "encrypted": true,
                "kmsKeyId":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        }
   ]
}
```

> **Important**
>
> Vous devez spécifier un type de `volume` ressource pour étiqueter les volumes Amazon EBS.

Performances des volumes Amazon EBS pour les tâches Fargate à la demande

Les IOPS et le débit du volume Amazon EBS de référence disponibles pour une tâche Fargate à la demande dépendent du nombre total d'unités de processeur que vous demandez pour la tâche. Si vous demandez 0,25, 0,5 ou 1 unité de processeur virtuelle (vCPU) pour votre tâche Fargate, nous vous recommandons de configurer un volume SSD à usage général `gp2` (`gp3`ou) ou un volume de disque dur (HDD) (ou). `st1` `sc1` Si vous demandez plus d'un vCPU pour votre tâche Fargate, les limites de performances de référence suivantes s'appliquent à un volume Amazon EBS associé à la tâche. Vous pouvez obtenir des performances EBS temporairement supérieures aux limites suivantes. Nous vous recommandons toutefois de planifier votre charge de travail en fonction de ces limites.

Unités de processeur demandées (en vCPUs)	IOPS Amazon EBS de référence (16 Kio d'E/S)	Débit de référence d'Amazon EBS (entrée MiBps, 128 Kio d'E/S)	Bande passante de référence (en Mbits/s)
2	3 000	75	360
4	5 000	120	1 150
8	10 000	250	2 300
16	15 000	500	4 500

> **Note**
>
> Lorsque vous configurez un volume Amazon EBS pour le joindre à une tâche Fargate, la limite de performance Amazon EBS pour la tâche Fargate est partagée entre le stockage éphémère de la tâche et le volume attaché.

Performances des volumes Amazon EBS pour les tâches EC2

Amazon EBS propose des types de volumes dont les caractéristiques de performance et le prix diffèrent, afin que vous puissiez adapter vos performances et vos coûts de stockage aux besoins de vos applications. Pour plus d'informations sur les performances, notamment les IOPS par volume

et le débit par volume, consultez les types de volumes Amazon EBS dans le guide de l'utilisateur d'Amazon Elastic Block Store.

Résolution des problèmes liés aux pièces jointes de volumes Amazon EBS aux tâches Amazon ECS

Vous devrez peut-être résoudre les problèmes ou vérifier l'attachement des volumes Amazon EBS aux tâches Amazon ECS.

Vérifier l'état de la connexion au volume

Vous pouvez utiliser le AWS Management Console pour consulter l'état de la pièce jointe d'un volume Amazon EBS à une tâche Amazon ECS. Si la tâche démarre et que la pièce jointe échoue, vous verrez également un motif de statut que vous pourrez utiliser pour résoudre le problème. Le volume créé sera supprimé et la tâche sera arrêtée. Pour plus d'informations sur les raisons liées au statut, consultezRaisons liées au statut de l'attachement d'un volume Amazon EBS aux tâches Amazon ECS.

Pour consulter l'état de la pièce jointe d'un volume et la raison du statut à l'aide de la console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster dans lequel votre tâche est exécutée. La page de détails du cluster apparaît.
3. Sur la page de détails du cluster, choisissez l'onglet Tâches.
4. Choisissez la tâche pour laquelle vous souhaitez consulter l'état des pièces jointes au volume. Vous devrez peut-être utiliser Filtrer le statut souhaité et choisir Arrêté si la tâche que vous souhaitez examiner s'est arrêtée.
5. Sur la page de détails de la tâche, choisissez l'onglet Volumes. Vous pourrez voir l'état de la pièce jointe du volume Amazon EBS sous État de la pièce jointe. Si le volume ne parvient pas à se joindre à la tâche, vous pouvez choisir l'état sous État de la pièce jointe pour afficher la cause de l'échec.

Vous pouvez également consulter l'état d'attachement au volume d'une tâche et la raison du statut associée à l'aide de l'DescribeTasksAPI.

Défaillances de service et de tâches

Vous pouvez rencontrer des défaillances de service ou de tâche qui ne sont pas spécifiques aux volumes Amazon EBS et qui peuvent affecter l'attachement des volumes. Pour plus d'informations, veuillez consulter la rubrique

- Messages d'événement de service
- Codes d'erreur des tâches arrêtées
- Raisons de défaillance de l'API

Raisons liées au statut de l'attachement d'un volume Amazon EBS aux tâches Amazon ECS

Utilisez la référence suivante pour résoudre les problèmes que vous pourriez rencontrer sous forme de raisons de statut AWS Management Console lorsque vous configurez des volumes Amazon EBS pour les associer à des tâches Amazon ECS. Pour plus d'informations sur la localisation de ces raisons d'état dans la console, consultezVérifier l'état de la connexion au volume.

ECS n'a pas pu assumer le rôle d'infrastructure ECS configuré « arn:aws:iam : :role/ ». *111122223333* *ecsInfrastructureRole* Vérifiez que le rôle transmis possède la bonne relation de confiance avec Amazon ECS

Cette raison de statut apparaît dans les scénarios suivants.

- Vous fournissez un rôle IAM sans que la politique de confiance nécessaire ne soit attachée. Amazon ECS ne peut pas accéder au rôle IAM d'infrastructure Amazon ECS que vous fournissez si le rôle ne dispose pas de la politique de confiance nécessaire. La tâche peut rester bloquée dans `DEPROVISIONING` cet état. Pour plus d'informations sur la politique de confiance nécessaire, consultezRôle IAM dans l'infrastructure Amazon ECS.
- Votre utilisateur IAM n'est pas autorisé à transmettre le rôle d'infrastructure Amazon ECS à Amazon ECS. La tâche peut rester bloquée dans `DEPROVISIONING` cet état. Pour éviter ce problème, vous pouvez associer l'`PassRole`autorisation à votre utilisateur. Pour de plus amples informations, veuillez consulter Rôle IAM dans l'infrastructure Amazon ECS.
- Votre rôle IAM ne dispose pas des autorisations nécessaires pour joindre un volume Amazon EBS. La tâche peut rester bloquée dans `DEPROVISIONING` cet état. Pour plus d'informations sur les autorisations spécifiques nécessaires pour associer des volumes Amazon EBS à des tâches, consultezRôle IAM dans l'infrastructure Amazon ECS.

> **Note**
>
> Ce message d'erreur peut également s'afficher en raison d'un retard dans la propagation des rôles. Si une nouvelle tentative d'utilisation du rôle après quelques minutes d'attente ne résout pas le problème, il se peut que vous ayez mal configuré la politique de confiance pour le rôle.

ECS n'a pas réussi à configurer le volume EBS. IdempotentParameterMismatchTrouvé « ; « Le jeton client que vous avez fourni est associé à une ressource déjà supprimée. Veuillez utiliser un autre jeton client. »

Les AWS KMS principaux scénarios suivants peuvent entraîner l'affichage d'un `IdempotentParameterMismatch` message :

- Vous spécifiez un ARN, un ID ou un alias de clé KMS non valide. Dans ce scénario, la tâche peut sembler être lancée avec succès, mais elle finit par échouer car elle AWS authentifie la clé KMS de manière asynchrone. Pour plus d'informations, consultez la section relative au chiffrement Amazon EBS dans le guide de EC2 l'utilisateur Amazon.
- Vous fournissez une clé gérée par le client qui ne dispose pas des autorisations permettant au rôle IAM de l'infrastructure Amazon ECS d'utiliser la clé pour le chiffrement. Pour éviter les problèmes d'autorisation liés aux politiques clés, consultez l'exemple de politique AWS KMS clé dans la section Chiffrement des données pour les volumes Amazon EBS.

Vous pouvez configurer Amazon EventBridge pour envoyer des événements de volume Amazon EBS et des événements de modification de l'état des tâches Amazon ECS à une cible, telle que CloudWatch des groupes Amazon. Vous pouvez ensuite utiliser ces événements pour identifier le problème spécifique lié aux clés gérées par le client qui a affecté l'attachement au volume. Pour plus d'informations, veuillez consulter la rubrique

- Comment créer un groupe de CloudWatch journaux à utiliser comme cible pour une EventBridge règle ? sur AWS Re:post.
- Événements de modification de l'état des tâches.
- EventBridge Événements Amazon pour Amazon EBS dans le guide de l'utilisateur Amazon EBS.

ECS a dépassé le délai imparti lors de la configuration du volume EBS attaché à votre tâche.

Les scénarios de format de système de fichiers suivants génèrent ce message.

- Le format du système de fichiers que vous spécifiez lors de la configuration n'est pas compatible avec le système d'exploitation de la tâche.
- Vous configurez un volume Amazon EBS pour qu'il soit créé à partir d'un instantané, et le format du système de fichiers de l'instantané n'est pas compatible avec le système d'exploitation de la tâche. Pour les volumes créés à partir d'un instantané, vous devez spécifier le même type de système de fichiers que celui utilisé par le volume lors de la création de l'instantané.

Vous pouvez utiliser les journaux de l'agent de conteneur Amazon ECS pour résoudre les problèmes liés à ce message pour les tâches de type EC2 lancement d'Amazon. Pour plus d'informations, consultez les sections Emplacements des fichiers journaux Amazon ECS et Amazon ECS Log Collector.

Utiliser les volumes Amazon EFS avec Amazon ECS

Amazon Elastic File System (Amazon EFS) fournit un stockage de fichiers simple et évolutif à utiliser avec vos tâches Amazon ECS. Avec Amazon EFS, la capacité de stockage est élastique. Elle augmente et diminue automatiquement au fil de vos ajouts et suppressions de fichiers. Vos applications peuvent disposer de l'espace de stockage qui leur est nécessaire, au moment où elles en ont besoin.

Vous pouvez utiliser le système de fichiers Amazon EFS avec Amazon ECS pour exporter des données du système de fichiers dans l'ensemble de votre flotte d'instances de conteneur. Ainsi, vos tâches ont accès au même stockage permanent, quelle que soit l'instance sur laquelle elles se retrouvent. Vos définitions de tâche doivent faire référence aux montages des volumes de l'instance de conteneur pour utiliser le système de fichiers.

Pour obtenir un didacticiel, consultez Configuration des systèmes de fichiers Amazon EFS pour Amazon ECS à l'aide de la console.

Considérations

Tenez compte des éléments suivants lorsque vous utilisez des volumes Amazon EFS :

- Pour les tâches utilisant le type de EC2 lancement, la prise en charge du système de fichiers Amazon EFS a été ajoutée en tant que version préliminaire publique avec la version AMI optimisée pour Amazon ECS `20191212` avec la version 1.35.0 de l'agent de conteneur. Toutefois, elle a été rendue généralement disponible avec l'AMI optimisée pour Amazon ECS version `20200319` et l'agent de conteneur version 1.38.0, qui contenait les fonctions de point d'accès Amazon EFS et d'autorisation IAM. Nous vous recommandons d'utiliser la version `20200319` de l'AMI optimisée pour Amazon ECS ou une version ultérieure pour utiliser ces fonctions. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.

> **Note**
>
> Si vous créez votre propre AMI, vous devez utiliser l'agent de conteneur 1.38.0 ou version ultérieure, `ecs-init` version 1.38.0-1 ou ultérieure, et exécuter les commandes suivantes

sur votre instance Amazon EC2 pour activer le plug-in de volume Amazon ECS. Les commandes varient selon que vous utilisez Amazon Linux 2 ou Amazon Linux comme image de base.

Amazon Linux 2

```
yum install amazon-efs-utils
systemctl enable --now amazon-ecs-volume-plugin
```

Amazon Linux

```
yum install amazon-efs-utils
sudo shutdown -r now
```

- Pour les tâches hébergées sur Fargate, les systèmes de fichiers Amazon EFS sont pris en charge sur la version 1.4.0 de la plateforme ou une version ultérieure (Linux). Pour de plus amples informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.
- Lorsque vous utilisez des volumes Amazon EFS pour des tâches hébergées sur Fargate, Fargate crée un conteneur de supervision chargé de la gestion du volume Amazon EFS. Le conteneur du superviseur utilise une petite partie de la mémoire et du processeur de la tâche. Le conteneur de supervision est visible lors de l'interrogation du point de terminaison de métadonnées de tâche version 4. En outre, il est visible dans CloudWatch Container Insights en tant que nom du conteneur`aws-fargate-supervisor`. Pour plus d'informations sur l'utilisation du type de EC2 lancement Amazon, consultezPoint de terminaison des métadonnées des tâches Amazon ECS, version 4. Pour plus d'informations sur l'utilisation du type de lancement Fargate, consultez. Point de terminaison de métadonnées de tâches Amazon ECS version 4 pour les tâches sur Fargate
- L'utilisation de volumes Amazon EFS ou la spécification d'une `EFSVolumeConfiguration` n'est pas prise en charge sur les instances externes.
- Nous vous recommandons de définir le paramètre `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` dans le fichier de configuration de l'agent sur une valeur inférieure à la valeur par défaut (environ une heure). Cette modification permet d'éviter l'expiration des informations d'identification de montage EFS et permet le nettoyage des montages qui ne sont pas utilisés. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

Utiliser les points d'accès Amazon EFS

Les points d'accès Amazon EFS sont des points d'entrée spécifiques à l'application dans un système de fichiers EFS pour gérer l'accès des applications aux jeux de données partagés. Pour de plus amples informations sur les points d'accès Amazon EFS et sur la façon de les contrôler, veuillez consulter Utilisation des points d'accès Amazon EFS dans le Guide de l'utilisateur Amazon Elastic File System.

Les points d'accès peuvent appliquer de manière forcée une identité d'utilisateur, y compris les groupes POSIX de l'utilisateur, pour toutes les demandes de système de fichiers effectuées via le point d'accès. Les points d'accès peuvent également imposer un répertoire racine différent au système de fichiers. Cela permet aux clients d'accéder uniquement aux données stockées dans le répertoire spécifié ou dans ses sous-répertoires.

> **Note**
>
> Lors de la création d'un point d'accès EFS, spécifiez un chemin d'accès sur le système de fichiers qui servira de répertoire racine. Lorsque vous faites référence au système de fichiers EFS avec un ID de point d'accès dans votre définition de tâche Amazon ECS, le répertoire racine doit être omis ou défini sur `/` ce qui permet d'appliquer le chemin défini sur le point d'accès EFS.

Vous pouvez utiliser le rôle IAM d'une tâche Amazon ECS pour imposer l'utilisation d'un point d'accès particulier par des applications spécifiques. En combinant des politiques IAM avec des points d'accès, vous pouvez fournir un accès sécurisé à des ensembles de données spécifiques pour vos applications. Pour plus d'informations sur l'utilisation des rôles IAM de tâche, consultez Rôle IAM de la tâche Amazon ECS.

Bonnes pratiques d'utilisation des volumes Amazon EFS avec Amazon ECS

Prenez note des recommandations de bonnes pratiques suivantes lorsque vous utilisez Amazon EFS avec Amazon ECS.

Contrôles de sécurité et d'accès pour les volumes Amazon EFS

Amazon EFS propose des fonctionnalités de contrôle d'accès que vous pouvez utiliser pour garantir que les données stockées dans un système de fichiers Amazon EFS sont sécurisées et accessibles uniquement depuis les applications qui en ont besoin. Vous pouvez sécuriser les données en activant

le chiffrement au repos et en transit. Pour en savoir plus, consultez Chiffrement des données dans Amazon EFS que vous trouverez dans le guide de l'utilisateur Amazon Elastic File System.

Outre le chiffrement des données, vous pouvez également utiliser Amazon EFS pour restreindre l'accès à un système de fichiers. Il existe trois manières de mettre en œuvre le contrôle d'accès dans EFS.

- Groupes de sécurité : avec les cibles de montage Amazon EFS, vous pouvez configurer un groupe de sécurité utilisé pour autoriser et refuser le trafic réseau. Vous pouvez configurer le groupe de sécurité attaché à Amazon EFS pour autoriser le trafic NFS (port 2049) en provenance du groupe de sécurité attaché à vos instances Amazon ECS ou, lorsque vous utilisez le mode `awsvpc` réseau, de la tâche Amazon ECS.
- IAM —Vous pouvez restreindre l'accès à un système de fichiers Amazon EFS à l'aide d'IAM. Une fois configurées, les tâches Amazon ECS nécessitent un rôle IAM pour accéder au système de fichiers afin de monter un système de fichiers EFS. Pour plus d'informations, consultez la section Utilisation d'IAM pour contrôler l'accès aux données du système de fichiers dans le manuel Amazon Elastic File System User Guide.

  Les politiques IAM peuvent également appliquer des conditions prédéfinies, telles que l'obligation pour un client d'utiliser le protocole TLS lorsqu'il se connecte à un système de fichiers Amazon EFS. Pour plus d'informations, consultez les clés de condition Amazon EFS pour les clients dans le guide de l'utilisateur Amazon Elastic File System.
- Points d'accès Amazon EFS : les points d'accès Amazon EFS sont des points d'entrée spécifiques à une application dans un système de fichiers Amazon EFS. Vous pouvez utiliser des points d'accès pour renforcer l'identité d'un utilisateur, y compris les groupes POSIX de l'utilisateur, pour toutes les demandes de système de fichiers effectuées via le point d'accès. Les points d'accès peuvent également imposer un répertoire racine différent au système de fichiers. Cela permet aux clients d'accéder uniquement aux données du répertoire spécifié ou de ses sous-répertoires.

Politiques IAM

Vous pouvez utiliser les politiques IAM pour contrôler l'accès au système de fichiers Amazon EFS.

Vous pouvez spécifier les actions suivantes pour les clients sur un système de fichiers à l'aide d'une stratégie de système de fichiers.

Action	Description
`elasticfilesystem:ClientMount`	Fournit un accès en lecture seule à un système de fichiers.
`elasticfilesystem:ClientWrite`	Fournit les droits d'écriture sur un système de fichiers.
`elasticfilesystem:ClientRootAccess`	Permet d'utiliser l'utilisateur root lors de l'accès à un système de fichiers.

Vous devez spécifier chaque action dans une politique. Les politiques peuvent être définies de la manière suivante :

- Basé sur le client : associez la politique au rôle de tâche

  Définissez l'option d'autorisation IAM lorsque vous créez la définition de tâche.
- Basé sur les ressources : attachez la politique au système de fichiers Amazon EFS

  Si la politique basée sur les ressources n'existe pas, par défaut, lors de la création du système de fichiers, l'accès est accordé à tous les principaux (*).

Lorsque vous définissez l'option d'autorisation IAM, nous fusionnons la politique associée au rôle de tâche et la politique basée sur les ressources Amazon EFS. L'option d'autorisation IAM transmet l'identité de la tâche (le rôle de la tâche) avec la politique à Amazon EFS. Cela permet à la politique basée sur les ressources Amazon EFS d'avoir un contexte pour l'utilisateur ou le rôle IAM spécifié dans la politique. Si vous ne définissez pas cette option, la politique au niveau des ressources Amazon EFS identifie l'utilisateur IAM comme étant « anonyme ».

Envisagez de mettre en œuvre les trois contrôles d'accès sur un système de fichiers Amazon EFS pour une sécurité maximale. Par exemple, vous pouvez configurer le groupe de sécurité attaché à un point de montage Amazon EFS pour autoriser uniquement le trafic NFS entrant provenant d'un groupe de sécurité associé à votre instance de conteneur ou à votre tâche Amazon ECS. En outre, vous pouvez configurer Amazon EFS pour exiger un rôle IAM pour accéder au système de fichiers, même si la connexion provient d'un groupe de sécurité autorisé. Enfin, vous pouvez utiliser les points d'accès Amazon EFS pour appliquer les autorisations des utilisateurs POSIX et spécifier des répertoires racines pour les applications.

L'extrait de définition de tâche suivant montre comment monter un système de fichiers Amazon EFS à l'aide d'un point d'accès.

```
"volumes": [
    {
      "efsVolumeConfiguration": {
        "fileSystemId": "fs-1234",
        "authorizationConfig": {
          "accessPointId": "fsap-1234",
          "iam": "ENABLED"
        },
        "transitEncryption": "ENABLED",
        "rootDirectory": ""
      },
      "name": "my-filesystem"
    }
]
```

Performances des volumes Amazon EFS

Amazon EFS propose deux modes de performance : les systèmes de I/O. General Purpose is suitable for latency-sensitive applications such as content management systems and CI/CD tools. In contrast, Max I/O fichiers General Purpose et Max conviennent aux charges de travail telles que l'analyse de données, le traitement multimédia et l'apprentissage automatique. Ces charges de travail doivent effectuer des opérations parallèles à partir de centaines, voire de milliers de conteneurs et nécessitent un débit agrégé et des IOPS les plus élevés possibles. Pour plus d'informations, consultez les modes de performance Amazon EFS dans le manuel Amazon Elastic File System User Guide.

Certaines charges de travail sensibles à la latence nécessitent à la fois les I/O niveaux les plus élevés fournis par le mode de I/O performance Max et les niveaux de latence plus faibles fournis par le mode de performance General Purpose. Pour ce type de charge de travail, nous vous recommandons de créer plusieurs systèmes de fichiers en mode de performance Usage général. Ainsi, vous pouvez répartir la charge de travail de votre application sur tous ces systèmes de fichiers, à condition que la charge de travail et les applications puissent la supporter.

Débit du volume Amazon EFS

Tous les systèmes de fichiers Amazon EFS sont associés à un débit mesuré qui est déterminé soit par le débit alloué pour les systèmes de fichiers utilisant le débit provisionné, soit par la quantité de données stockées dans la norme EFS ou dans la classe de stockage One Zone pour les systèmes de

fichiers utilisant le débit en rafale. Pour plus d'informations, consultez la section Comprendre le débit mesuré dans le guide de l'utilisateur d'Amazon Elastic File System.

Le mode de débit par défaut pour les systèmes de fichiers Amazon EFS est le mode rafale. Avec le mode rafale, le débit disponible pour un système de fichiers augmente ou diminue au fur et à mesure que le système de fichiers grandit. Étant donné que les charges de travail basées sur des fichiers augmentent généralement, nécessitant des niveaux de débit élevés pendant un certain temps et des niveaux de débit inférieurs le reste du temps, Amazon EFS est conçu pour fonctionner en rafale afin de permettre des niveaux de débit élevés pendant un certain temps. En outre, étant donné que de nombreuses charges de travail sont gourmandes en lecture, les opérations de lecture sont mesurées selon un ratio de 1:3 par rapport aux autres opérations NFS (comme l'écriture).

Tous les systèmes de fichiers Amazon EFS fournissent une performance de référence constante de 50 MB/s % pour chaque To de stockage Amazon EFS Standard ou Amazon EFS One Zone. Tous les systèmes de fichiers (quelle que soit leur taille) peuvent atteindre 100 MB/s. File systems with more than 1TB of EFS Standard or EFS One Zone storage can burst to 100 MB/s for each TB. Because read operations are metered at a 1:3 ratio, you can drive up to 300 MiBs/s pour chaque TiB de débit de lecture. Lorsque vous ajoutez des données à votre système de fichiers, le débit maximal disponible pour le système de fichiers évolue de manière linéaire et automatique en fonction de votre stockage dans la classe de stockage Amazon EFS Standard. Si vous avez besoin d'un débit supérieur à celui que vous pouvez atteindre avec la quantité de données stockées, vous pouvez configurer le débit provisionné en fonction de la quantité spécifique requise par votre charge de travail.

Le débit du système de fichiers est partagé entre toutes les EC2 instances Amazon connectées à un système de fichiers. Par exemple, un système de fichiers de 1 To capable d'atteindre un débit MB/s de 100 % peut en générer 100 à MB/s partir d'une seule EC2 instance Amazon capable de générer 10 Mo/s chacune. Pour plus d'informations, consultez la section relative aux performances d'Amazon EFS dans le manuel Amazon Elastic File System User Guide.

Optimisation des coûts pour les volumes Amazon EFS

Amazon EFS simplifie le dimensionnement du stockage pour vous. Les systèmes de fichiers Amazon EFS augmentent automatiquement au fur et à mesure que vous ajoutez des données. En particulier avec le mode Amazon EFS Bursting Throughput, le débit d'Amazon EFS augmente en fonction de la taille de votre système de fichiers dans la classe de stockage standard. Pour améliorer le débit sans payer de frais supplémentaires pour le débit alloué sur un système de fichiers EFS, vous pouvez partager un système de fichiers Amazon EFS avec plusieurs applications. À l'aide des points d'accès Amazon EFS, vous pouvez implémenter l'isolation du stockage dans les systèmes de fichiers

Amazon EFS partagés. Ainsi, même si les applications partagent toujours le même système de fichiers, elles ne peuvent accéder aux données que si vous les autorisez.

Au fur et à mesure que vos données augmentent, Amazon EFS vous aide à déplacer automatiquement les fichiers rarement consultés vers une classe de stockage inférieure. La classe de stockage Amazon EFS Standard-Infrequent Access (IA) réduit les coûts de stockage pour les fichiers auxquels on n'accède pas tous les jours. Cela se fait sans sacrifier la haute disponibilité, la durabilité élevée, l'élasticité et l'accès au système de fichiers POSIX fournis par Amazon EFS. Pour plus d'informations, consultez les classes de stockage EFS dans le manuel Amazon Elastic File System User Guide.

Envisagez d'utiliser les politiques de cycle de vie d'Amazon EFS pour économiser automatiquement de l'argent en transférant les fichiers rarement consultés vers le stockage Amazon EFS IA. Pour plus d'informations, consultez Gestion du cycle de vie Amazon EFS dans le guide de l'utilisateur Amazon Elastic File System.

Lorsque vous créez un système de fichiers Amazon EFS, vous pouvez choisir si Amazon EFS réplique vos données sur plusieurs zones de disponibilité (standard) ou les stocke de manière redondante dans une seule zone de disponibilité. La classe de stockage Amazon EFS One Zone permet de réduire considérablement les coûts de stockage par rapport aux classes de stockage Amazon EFS Standard. Envisagez d'utiliser la classe de stockage Amazon EFS One Zone pour les charges de travail ne nécessitant pas de résilience multi-AZ. Vous pouvez réduire davantage le coût du stockage Amazon EFS One Zone en déplaçant les fichiers rarement consultés vers Amazon EFS One Zone-Infrequent Access. Pour plus d'informations, veuillez consulter Amazon EFS Infrequent Access.

Protection des données des volumes Amazon EFS

Amazon EFS stocke vos données de manière redondante dans plusieurs zones de disponibilité pour les systèmes de fichiers à l'aide de classes de stockage standard. Si vous sélectionnez les classes de stockage Amazon EFS One Zone, vos données sont stockées de manière redondante dans une seule zone de disponibilité. En outre, Amazon EFS est conçu pour fournir une durabilité de 99,999999999 % (11 9) sur une année donnée.

Comme dans tout environnement, il est recommandé de disposer d'une sauvegarde et de mettre en place des mesures de protection contre les suppressions accidentelles. Pour les données Amazon EFS, cette bonne pratique inclut une sauvegarde fonctionnelle et régulièrement testée à l'aide de AWS Backup. Les systèmes de fichiers utilisant les classes de stockage Amazon EFS One Zone sont configurés pour sauvegarder automatiquement les fichiers par défaut lors de la création du système

de fichiers, sauf si vous choisissez de désactiver cette fonctionnalité. Pour plus d'informations, consultez la section Sauvegarde des systèmes de fichiers EFS dans le manuel Amazon Elastic File System User Guide.

Spécifier un système de fichiers Amazon EFS dans une définition de tâche Amazon ECS

Pour utiliser des volumes de système de fichiers Amazon EFS pour vos conteneurs, vous devez spécifier les configurations de volume et de point de montage dans votre définition de tâche. L'extrait de JSON de définition de tâche indiqué ci-dessous illustre la syntaxe des objets `volumes` et `mountPoints` pour un conteneur.

```
{
    "containerDefinitions": [
        {
            "name": "container-using-efs",
            "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
            "entryPoint": [
                "sh",
                "-c"
            ],
            "command": [
                "ls -la /mount/efs"
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEfsVolume",
                    "containerPath": "/mount/efs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEfsVolume",
            "efsVolumeConfiguration": {
                "fileSystemId": "fs-1234",
                "rootDirectory": "/path/to/my/data",
                "transitEncryption": "ENABLED",
                "transitEncryptionPort": integer,
                "authorizationConfig": {
                    "accessPointId": "fsap-1234",
                    "iam": "ENABLED"
```

```
                }
            }
        }
    ]
}
```

`efsVolumeConfiguration`

Type : objet

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Amazon EFS.

`fileSystemId`

Type : String

Obligatoire : oui

ID du système de fichiers Amazon EFS à utiliser.

`rootDirectory`

Type : chaîne

Obligatoire : non

Répertoire du système de fichiers Amazon EFS à monter en tant que répertoire racine à l'intérieur de l'hôte. Si ce paramètre est omis, la racine du volume Amazon EFS est utilisée. La spécification de `/` a le même effet que l'omission de ce paramètre.

> **Important**
>
> Si un point d'accès EFS est spécifié dans `authorizationConfig`, le paramètre de répertoire racine doit être omis ou défini sur `/` ce qui permet d'appliquer le chemin défini sur le point d'accès EFS.

`transitEncryption`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Indique si vous souhaitez activer ou non le chiffrement des données Amazon EFS en transit entre l'hôte Amazon ECS et le serveur Amazon EFS. Si l'autorisation Amazon EFS IAM est utilisée, le chiffrement de transit doit être activé. Si ce paramètre est omis, la valeur par défaut `DISABLED` est utilisée. Pour plus d'informations, consultez Chiffrement des données en transit dans le Guide de l'utilisateur Amazon Elastic File System.

`transitEncryptionPort`

Type : entier

Obligatoire : non

Port à utiliser lors de l'envoi de données chiffrées entre l'hôte Amazon ECS et le serveur Amazon EFS. Si vous ne spécifiez pas de port de chiffrement en transit, il utilise la stratégie de sélection de port adoptée par l'assistant de montage Amazon EFS. Pour plus d'informations, consultez Assistant de montage EFS dans le Guide de l'utilisateur Amazon Elastic File System User.

`authorizationConfig`

Type : objet

Obligatoire : non

Détails de configuration des autorisations pour le système de fichiers Amazon EFS.

`accessPointId`

Type : chaîne

Obligatoire : non

ID du point d'accès à utiliser. Si un point d'accès est spécifié, la valeur du répertoire racine dans `efsVolumeConfiguration` doit être omise ou définie sur `/` qui applique le chemin défini sur le point d'accès EFS. Si un point d'accès est utilisé, le chiffrement de transit doit être activé dans `EFSVolumeConfiguration`. Pour plus d'informations, consultez Utilisation des points d'accès Amazon EFS dans le Guide de l'utilisateur Amazon Elastic File System.

`iam`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Indique s'il faut ou non utiliser le rôle IAM de tâche Amazon ECS spécifié dans une définition de tâche lors du montage du système de fichiers Amazon EFS. Si cette option est activée, le chiffrement en transit doit être activé dans la configuration `EFSVolumeConfiguration`. Si ce paramètre est omis, la valeur par défaut `DISABLED` est utilisée. Pour plus d'informations, consultez Rôles IAM pour les tâches.

Configuration des systèmes de fichiers Amazon EFS pour Amazon ECS à l'aide de la console

Découvrez comment utiliser les systèmes de fichiers Amazon Elastic File System (Amazon EFS) avec Amazon ECS.

Étape 1 : Créer un cluster Amazon ECS

Pour créer un cluster Amazon ECS, effectuez les étapes suivantes.

Pour créer un nouveau cluster (console Amazon ECS)

Avant de commencer, attribuez l'autorisation IAM adéquate. Pour de plus amples informations, veuillez consulter the section called “Exemples de clusters Amazon ECS”.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Cluster configuration (Configuration de cluster), dans Cluster name (Nom du cluster), saisissez `EFS-tutorial`.
6. (Facultatif) Pour modifier le VPC et les sous-réseaux d'où vos tâches et services se lancent, sous Networking (Réseaux), effectuez l'une des opérations suivantes :
   - Pour supprimer un sous-réseau, sous Subnets (Sous-réseaux), choisissez X pour chaque sous-réseau que vous souhaitez supprimer.
   - Pour passer à un VPC autre celui par défaut, sous VPC, choisissez un VPC existant, puis sous Subnets (Sous-réseaux), sélectionnez chaque sous-réseau.

7. Pour ajouter des EC2 instances Amazon à votre cluster, développez l'infrastructure, puis sélectionnez EC2les instances Amazon. Ensuite, configurez le groupe Auto Scaling qui agit en tant que fournisseur de capacité :
   - Pour créer un groupe Auto Scaling, depuis Auto Scaling group (ASG) (Groupe Auto Scaling [ASG]), sélectionnez Create new group (Créer un nouveau groupe), puis fournissez les détails suivants sur le groupe :
     - Pour Operating system/Architecture (Système d'exploitation/Architecture), choisissez Amazon Linux 2.
     - EC2 Par type d'instance, choisissez`t2.micro`.

       Pour SSH key pair (Paire de clés SSH), choisissez la paire qui prouve votre identité lorsque vous connectez à l'instance.
     - Dans le champ Capacité, entrez 1.
8. Choisissez Créer.

Étape 2 : créer un groupe de sécurité pour les EC2 instances Amazon et le système de fichiers Amazon EFS

Au cours de cette étape, vous créez un groupe de sécurité pour vos EC2 instances Amazon qui autorise le trafic réseau entrant sur le port 80 et votre système de fichiers Amazon EFS qui autorise l'accès entrant depuis vos instances de conteneur.

Créez un groupe de sécurité pour vos EC2 instances Amazon avec les options suivantes :

- Nom du groupe de sécurité : un nom unique pour votre groupe de sécurité.
- VPC : le VPC identifié précédemment pour votre cluster.
- Règle entrante
  - Type : HTTP.
  - Source : 0.0.0.0/0.

Créez un groupe de sécurité pour votre système de fichiers Amazon EFS avec les options suivantes :

- Nom du groupe de sécurité : un nom unique pour votre groupe de sécurité. Par exemple, `EFS-access-for-sg-`*`dc025fa2`*.
- VPC : le VPC identifié précédemment pour votre cluster.

- Règle entrante
  - Type : NFS.
  - Source : personnalisée avec l'ID du groupe de sécurité que vous avez créé pour vos instances.

Pour plus d'informations sur la création d'un groupe de sécurité, consultez la section Créer un groupe de sécurité pour votre EC2 instance Amazon dans le guide de EC2 l'utilisateur Amazon.

Étape 3 : Créer un système de fichiers Amazon EFS

Dans cette étape, vous créez un système de fichiers Amazon EFS.

Pour créer un système de fichiers Amazon EFS pour des tâches Amazon ECS.

1. Ouvrez la console Amazon Elastic File System à l'adresse https://console.aws.amazon.com/efs/.
2. Choisissez Create file system (Créer un système de fichiers).
3. Saisissez un nom pour votre système de fichiers, puis choisissez le VPC dans lequel vos instances de conteneur sont hébergées. Par défaut, chaque sous-réseau du VPC spécifié reçoit une cible de montage qui utilise le groupe de sécurité par défaut de ce VPC. Choisissez ensuite Personnaliser.

   > **Note**
   >
   > Ce didacticiel part du principe que votre système de fichiers Amazon EFS, votre cluster Amazon ECS, vos instances de conteneur et vos tâches se trouvent dans le même VPC. Pour plus d'informations sur le montage d'un système de fichiers à partir d'un autre VPC, consultez la section Procédure pas à pas : monter un système de fichiers à partir d'un autre VPC dans le guide de l'utilisateur Amazon EFS.

4. Sur la page Paramètres du système de fichiers, configurez les paramètres facultatifs, puis sous Paramètres de performance, choisissez le mode de débit Transmission en rafales pour votre système de fichiers. Après avoir configuré les paramètres, sélectionnez Suivant.

   a. (Facultatif) Ajoutez des étiquettes pour votre système de fichiers. Par exemple, vous pouvez spécifier un nom unique pour le système de fichiers en entrant ce nom dans la colonne Valeur en regard de la clé Nom.

   b. (Facultatif) Activez la gestion du cycle de vie pour économiser de l'argent sur un stockage rarement utilisé. Pour en savoir plus, consultez la section Gestion du cycle de vie EFS du Amazon Elastic File System User Guide.

   c. (Facultatif) Activez le chiffrement. Cochez la case pour activer le chiffrement de votre système de fichiers Amazon EFS au repos.

5. Sur la page Accès réseau, sous Monter les cibles, remplacez la configuration de groupe de sécurité existante de chaque zone de disponibilité par le groupe de sécurité que vous avez créé pour le système de fichiers dans Étape 2 : créer un groupe de sécurité pour les EC2 instances Amazon et le système de fichiers Amazon EFS, puis choisissez Suivant.
6. Il n'est pas nécessaire de configurer la Stratégie de système de fichiers pour ce didacticiel. Vous pouvez donc ignorer la section en choisissant Suivant.
7. Vérifiez les options de votre système de fichiers et choisissez Créer pour terminer le processus.
8. Sur l'écran Systèmes de fichiers, enregistrez l'ID du système de fichiers. À l'étape suivante, vous référencerez cette valeur dans votre définition de tâche Amazon ECS.

Étape 4 : Ajouter du contenu au système de fichiers Amazon EFS

Au cours de cette étape, vous montez le système de fichiers Amazon EFS sur une EC2 instance Amazon et vous y ajoutez du contenu. Cette opération servira à tester la nature persistante des données dans ce didacticiel. Lorsque vous utilisez cette fonction, vous avez normalement votre application ou une autre méthode d'écriture de données dans votre système de fichiers Amazon EFS.

Pour créer une EC2 instance Amazon et monter le système de fichiers Amazon EFS

1. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
2. Choisissez Launch Instances (Lancer les instances).
3. Sous Images d'applications et de systèmes d'exploitation (Amazon Machine Image), sélectionnez l'AMI Amazon Linux 2 (HVM).
4. Sous Type d'instance, conservez le type d'instance par défaut `t2.micro`.
5. Sous Paire de clés (connexion), sélectionnez une paire de clés pour l'accès SSH à l'instance.
6. Sous Paramètres réseau, sélectionnez le VPC que vous avez spécifié pour votre système de fichiers Amazon EFS et votre cluster Amazon ECS. Sélectionnez un sous-réseau et le groupe de sécurité d'instance créé dans Étape 2 : créer un groupe de sécurité pour les EC2 instances Amazon et le système de fichiers Amazon EFS. Configurez le groupe de sécurité de l'instance. Assurez-vous que Attribuer automatiquement l'adresse IP publique est activé.
7. Sous Configurer le stockage, cliquez sur le bouton Modifier pour les systèmes de fichiers, puis choisissez EFS. Sélectionnez le système de fichiers créé dans Étape 3 : Créer un système de

fichiers Amazon EFS. Vous pouvez éventuellement modifier le point du montage ou conserver la valeur par défaut.

> **Important**
>
> Vous devez sélectionner un sous-réseau avant de pouvoir ajouter un système de fichiers à l'instance.

8. Désactivez Créer et associer automatiquement des groupes de sécurité. Laissez l'autre case cochée. Choisissez Add shared file system (Ajouter un système de fichiers partagé).
9. Sous Advanced Details (Détails avancés), assurez-vous que le script de données utilisateur est renseigné automatiquement avec les étapes de montage du système de fichiers Amazon EFS.
10. Sous Récapitulatif, assurez-vous que le Nombre d'instances est égal à 1. Choisissez Launch instance (Lancer une instance).
11. Sur la page Lancer une instance, choisissez Afficher toutes les instances pour afficher l'état de vos instances. Initialement, le statut d'État de l'instance est `PENDING`. Une fois que l'état est passé à `RUNNING` et que l'instance a réussi toutes les vérifications de statut, celle-ci est prête à être utilisée.

À présent, vous vous connectez à l' EC2 instance Amazon et ajoutez du contenu au système de fichiers Amazon EFS.

Pour vous connecter à l' EC2 instance Amazon et ajouter du contenu au système de fichiers Amazon EFS

1. Connectez-vous en SSH à l' EC2 instance Amazon que vous avez créée. Pour plus d'informations, consultez Connect to your Linux instance using SSH dans le Amazon EC2 User Guide.
2. Dans la fenêtre du terminal, exécutez la commande df -T pour vérifier que le système de fichiers Amazon EFS est bien monté. Dans la sortie suivante, nous avons mis en évidence le montage du système de fichiers Amazon EFS.

```
$ df -T
Filesystem     Type            1K-blocks    Used        Available Use% Mounted on
devtmpfs       devtmpfs           485468       0           485468   0% /dev
tmpfs          tmpfs              503480       0           503480   0% /dev/shm
tmpfs          tmpfs              503480     424           503056   1% /run
```

```
tmpfs           tmpfs               503480       0           503480   0% /sys/fs/
cgroup
/dev/xvda1      xfs                8376300 1310952          7065348  16% /
127.0.0.1:/     nfs4     9007199254739968       0 9007199254739968   0% /mnt/efs/fs1
tmpfs           tmpfs               100700       0           100700   0% /run/
user/1000
```

3. Accédez au répertoire dans lequel le système de fichiers Amazon EFS est monté. Dans l'exemple ci-dessus, il s'agit de `/mnt/efs/fs1`.
4. Créez un fichier nommé `index.html` avec le contenu suivant:

```
<html>
    <body>
        <h1>It Works!</h1>
        <p>You are using an Amazon EFS file system for persistent container
 storage.</p>
    </body>
</html>
```

Étape 5 : Créer une définition de tâche

La définition de tâche suivante crée un volume de données nommé `efs-html`. Le conteneur `nginx` monte le volume de données de l'hôte à la racine NGINX, `/usr/share/nginx/html`.

Pour créer une nouvelle définition de tâche à l'aide de la console Amazon ECS

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
4. Dans la zone de l'éditeur JSON, copiez et collez le texte JSON suivant, en remplaçant `fileSystemId` par l'ID de votre système de fichiers Amazon EFS.

```
{
    "containerDefinitions": [
        {
            "memory": 128,
            "portMappings": [
                {
                    "hostPort": 80,
```

```
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "mountPoints": [
                {
                    "containerPath": "/usr/share/nginx/html",
                    "sourceVolume": "efs-html"
                }
            ],
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest"
        }
    ],
    "volumes": [
        {
            "name": "efs-html",
            "efsVolumeConfiguration": {
                "fileSystemId": "fs-1324abcd",
                "transitEncryption": "ENABLED"
            }
        }
    ],
    "family": "efs-tutorial",
    "executionRoleArn":"arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
}
```

> **Note**
>
> Le rôle IAM d'exécution de tâches Amazon ECS ne nécessite aucune autorisation spécifique liée à Amazon EFS pour monter un système de fichiers Amazon EFS. Par défaut, s'il n'existe aucune politique basée sur les ressources Amazon EFS, l'accès est accordé à tous les principaux (*) lors de la création du système de fichiers.
> Le rôle de tâche Amazon ECS n'est requis que si « l'autorisation EFS IAM » est activée dans la définition de tâche Amazon ECS. Lorsque cette option est activée, l'identité du rôle de la tâche doit être autorisée à accéder au système de fichiers Amazon EFS conformément à la politique basée sur les ressources Amazon EFS, et l'accès anonyme doit être désactivé.

5. Choisissez Créer.

Étape 6 : Exécuter une tâche et afficher les résultats

Maintenant que votre système de fichiers Amazon EFS est créé et qu'il existe du contenu web pour le conteneur NGINX à diffuser, vous pouvez exécuter une tâche à l'aide de la définition de tâche que vous avez créée. Le serveur web NGINX diffuse votre page HTML simple. Si vous mettez à jour le contenu de votre système de fichiers Amazon EFS, ces changements sont propagés vers tous les conteneurs qui ont également monté ce système de fichiers.

La tâche s'exécute dans le sous-réseau que vous avez défini pour le cluster.

Pour exécuter une tâche et afficher les résultats à l'aide de la console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, sélectionnez le cluster dans lequel la tâche autonome doit s'exécuter.

   Déterminez la ressource à partir de laquelle vous lancez le service.

Pour démarrer un service à partir de	Étapes	
Clusters	a. Sur la page Clusters, sélectionnez le cluster dans lequel créer le service. b. Sous l'onglet Tasks (Tâches), choisissez Run new task (Exécuter une nouvelle tâche).	
Type de lancement	a. Sur la page Task (Tâche), choisissez la définition de tâche. b. S'il y a plusieurs révisions, sélectionnez-en une. c. Choisissez Create (Créer), Exécuter la tâche (Exécuter la tâche).	

3. (Facultatif) Choisissez comment votre tâche planifiée est distribuée dans votre infrastructure de cluster. Développer Compute configuration (Configuration de calcul), puis procédez comme suit :

Méthode de distribution	Étapes
Type de lancement	a. Dans Compute options (Options de calcul), sélectionnez Launch type (Type de lancement). b. Dans Type de lancement, sélectionnez EC2.

4. Pour Application type (Type d'application), choisissez Task (Tâche).
5. Pour Task Definition (Définition de tâche), choisissez la définition de tâche `efs-tutorial` que vous avez créée précédemment.
6. Pour Desired tasks (Tâches souhaitées), saisissez 1.
7. Choisissez Créer.
8. Sur la page Cluster, choisissez Infrastructure.
9. Sous Instances de conteneur, sélectionnez l'instance de conteneur à laquelle vous connecter.
10. Sur la page Container Instance (Instance de conteneur), sous Networking (Mise en réseau), enregistrez Public IP (IP publique) pour votre instance.
11. Ouvrez un navigateur et saisissez l'adresse IP publique. Vous devez voir le message suivant :

```
It works!
You are using an Amazon EFS file system for persistent container storage.
```

> **Note**
>
> Si le message ne s'affiche pas, assurez-vous que le groupe de sécurité de votre instance de conteneur autorise le trafic réseau entrant sur le port 80 et que le groupe de sécurité de votre système de fichiers autorise l'accès entrant depuis l'instance de conteneur.

Utilisation FSx pour les volumes de serveurs de fichiers Windows avec Amazon ECS

FSx pour Windows File Server fournit des serveurs de fichiers Windows entièrement gérés, soutenus par un système de fichiers Windows. Lorsque vous utilisez FSx for Windows File Server avec ECS, vous pouvez fournir à vos tâches Windows un stockage de fichiers statique persistant, distribué, partagé et statique. Pour plus d'informations, voir Qu'est-ce que FSx pour le serveur de fichiers Windows ? .

> **Note**
>
> EC2 les instances qui utilisent l'AMI complète Windows Server 2016 optimisée pour Amazon ECS ne prennent pas en charge FSx les volumes de tâches ECS du serveur de fichiers Windows.
> Vous ne pouvez pas FSx les utiliser pour les volumes du serveur de fichiers Windows dans une configuration de conteneurs Windows sur Fargate. Au lieu de cela, vous pouvez modifier les conteneurs pour les monter au démarrage.

Vous pouvez utiliser Windows File Server FSx pour déployer des charges de travail Windows nécessitant un accès à un espace de stockage externe partagé, à un stockage régional à haute disponibilité ou à un stockage à haut débit. Vous pouvez monter un ou plusieurs volumes de système de fichiers FSx pour Windows File Server sur un conteneur Amazon ECS qui s'exécute sur une instance Windows Amazon ECS. Vous pouvez partager FSx des volumes de système de fichiers pour Windows File Server entre plusieurs conteneurs Amazon ECS dans le cadre d'une seule tâche Amazon ECS.

Pour permettre l'utilisation de FSx for Windows File Server avec ECS, incluez l'ID du système de fichiers FSx pour Windows File Server et les informations associées dans une définition de tâche. Il s'agit de l'extrait de code JSON de définition de tâche suivant. Avant de créer et d'exécuter une définition de tâche, vous avez besoin des éléments suivants.

- Une EC2 instance Windows ECS jointe à un domaine valide. Il peut être hébergé par un AWS Directory Service for Microsoft Active DirectoryActive Directory sur site ou par un Active Directory auto-hébergé sur Amazon. EC2
- Paramètre AWS Secrets Manager secret ou Systems Manager contenant les informations d'identification utilisées pour rejoindre le domaine Active Directory et attacher le FSx système de fichiers Windows File Server. Les valeurs d'informations d'identification sont les informations

d'identification de nom et de mot de passe que vous avez entrées lors de la création de l'Active Directory.

Pour voir un didacticiel connexe, consultez Découvrez comment configurer FSx les systèmes de fichiers Windows File Server pour Amazon ECS.

Considérations

Tenez compte des points suivants lors de l' FSx utilisation de volumes de serveurs de fichiers Windows :

- FSx pour Windows File Server avec Amazon ECS ne prend en charge que les EC2 instances Windows Amazon. Les EC2 instances Amazon Linux ne sont pas prises en charge.
- FSx pour Windows File Server avec Amazon ECS n'est pas pris en charge AWS Fargate.
- FSx pour Windows File Server avec Amazon ECS en mode `awsvpc` réseau, nécessite une version `1.54.0` ou une version ultérieure de l'agent de conteneur.
- Le nombre maximal de lettres de lecteur pouvant être utilisées pour une tâche Amazon ECS est de 23. Une lettre de lecteur est attribuée FSx à chaque tâche comportant un volume destiné au serveur de fichiers Windows.
- Par défaut, le temps de nettoyage des ressources de tâches est de trois heures après la fin de la tâche. Même si aucune tâche ne l'utilise, un mappage de fichier créé par une tâche persiste pendant trois heures. La durée de nettoyage par défaut peut être configurée à l'aide de la variable d'environnement Amazon ECS `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.
- Les tâches s'exécutent généralement uniquement dans le même VPC que le système FSx de fichiers Windows File Server. Cependant, il est possible de bénéficier d'un support inter-VPC s'il existe une connectivité réseau établie entre le VPC du cluster Amazon ECS et le système de fichiers FSx pour Windows File Server via le peering VPC.
- Vous contrôlez l'accès à un système de fichiers FSx pour Windows File Server au niveau du réseau en configurant les groupes de sécurité VPC. Seules les tâches hébergées sur EC2 des instances associées au domaine Active Directory avec des groupes de sécurité Active Directory correctement configurés peuvent accéder au partage de fichiers FSx pour Windows File Server. Si les groupes de sécurité sont mal configurés, Amazon ECS ne parvient pas à lancer la tâche avec le message d'erreur suivant : `unable to mount file system` *`fs-id`* »
- FSx pour Windows File Server est intégré à AWS Identity and Access Management (IAM) pour contrôler les actions que vos utilisateurs et groupes IAM peuvent effectuer spécifiquement FSx

pour les ressources du serveur de fichiers Windows. Avec l'autorisation du client, les clients peuvent définir des rôles IAM qui autorisent ou refusent l'accès à FSx des systèmes de fichiers spécifiques à Windows File Server, exigent éventuellement un accès en lecture seule et autorisent ou interdisent au client l'accès root au système de fichiers. Pour plus d'informations, consultez la section Sécurité dans le guide de l'utilisateur Amazon FSx Windows.

Bonnes pratiques d'utilisation FSx pour Windows File Server avec Amazon ECS

Prenez note des recommandations de bonnes pratiques suivantes lorsque vous utilisez FSx un serveur de fichiers Windows avec Amazon ECS.

Sécurité et contrôles d'accès FSx pour Windows File Server

FSx pour Windows File Server propose les fonctionnalités de contrôle d'accès suivantes que vous pouvez utiliser pour garantir que les données stockées dans un système de fichiers FSx pour Windows File Server sont sécurisées et accessibles uniquement depuis les applications qui en ont besoin.

Chiffrement des données FSx pour les volumes du serveur de fichiers Windows

FSx pour Windows File Server prend en charge deux formes de chiffrement pour les systèmes de fichiers. Il s'agit du chiffrement des données en transit et du chiffrement au repos. Le chiffrement des données en transit est pris en charge sur les partages de fichiers mappés sur une instance de conteneur compatible avec le protocole SMB 3.0 ou une version ultérieure. Le chiffrement des données au repos est automatiquement activé lors de la création d'un système de FSx fichiers Amazon. Amazon chiffre FSx automatiquement les données en transit à l'aide du chiffrement SMB lorsque vous accédez à votre système de fichiers sans avoir à modifier vos applications. Pour plus d'informations, consultez la section Chiffrement des données sur Amazon FSx dans le guide de l'utilisateur du serveur de fichiers Amazon FSx pour Windows.

Utiliser Windows ACLs pour le contrôle d'accès au niveau des dossiers

L' EC2 instance Amazon Windows accède aux partages de FSx fichiers Amazon à l'aide des informations d'identification Active Directory. Il utilise des listes de contrôle d'accès Windows standard (ACLs) pour un contrôle d'accès précis au niveau des fichiers et des dossiers. Vous pouvez créer plusieurs informations d'identification, chacune correspondant à un dossier spécifique du partage correspondant à une tâche spécifique.

Dans l'exemple suivant, la tâche a accès au dossier à l'`App01`aide d'un identifiant enregistré dans Secrets Manager. Son Amazon Resource Name (ARN) est`1234`.

```
"rootDirectory": "\\path\\to\\my\\data\App01",
"credentialsParameter": "arn-1234",
"domain": "corp.fullyqualified.com",
```

Dans un autre exemple, une tâche a accès au dossier à l'App02aide d'un identifiant enregistré dans le Secrets Manager. Son ARN est 6789.

```
"rootDirectory": "\\path\\to\\my\\data\App02",
"credentialsParameter": "arn-6789",
"domain": "corp.fullyqualified.com",
```

Spécifiez un système de fichiers FSx pour Windows File Server dans une définition de tâche Amazon ECS

À utiliser FSx pour les volumes du système de fichiers Windows File Server pour vos conteneurs, spécifiez les configurations de volume et de point de montage dans votre définition de tâche. L'extrait de JSON de définition de tâche indiqué ci-dessous illustre la syntaxe des objets `volumes` et `mountPoints` pour un conteneur.

```
{
    "containerDefinitions": [
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [],
            "command": ["New-Item -Path C:\\fsx-windows-dir\\index.html -ItemType file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>It Works!</h2> <p>You are using Amazon
 FSx for Windows File Server file system for persistent container storage.</p>' -
Force"],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": false,
            "name": "container1",
            "mountPoints": [
                {
```

```
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ]
        },
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [
                {
                    "hostPort": 443,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": ["Remove-Item -Recurse C:\\inetpub\\wwwroot\\* -Force; Start-
Sleep -Seconds 120; Move-Item -Path C:\\fsx-windows-dir\\index.html -Destination C:\
\inetpub\\wwwroot\\index.html -Force; C:\\ServiceMonitor.exe w3svc"],
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": true,
            "name": "container2"
        }
    ],
    "family": "fsx-windows",
    "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
    "volumes": [
        {
            "name": "fsx-windows-dir",
            "fsxWindowsFileServerVolumeConfiguration": {
                "fileSystemId": "fs-0eeb5730b2EXAMPLE",
                "authorizationConfig": {
```

```
                    "domain": "example.com",
                    "credentialsParameter": "arn:arn-1234"
                },
                "rootDirectory": "share"
            }
        }
    ]
}
```

`FSxWindowsFileServerVolumeConfiguration`

Type : objet

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez FSx le système de fichiers Windows File Server pour le stockage des tâches.

`fileSystemId`

Type : String

Obligatoire : oui

L' FSx identifiant du système de fichiers Windows File Server à utiliser.

`rootDirectory`

Type : String

Obligatoire : oui

Le répertoire du système FSx de fichiers Windows File Server à monter en tant que répertoire racine sur l'hôte.

`authorizationConfig`

`credentialsParameter`

Type : String

Obligatoire : oui

Les options d'informations d'identification d'autorisation :

- Amazon Resource Name (ARN) pour un secret Secrets Manager.
- Amazon Resource Name (ARN) pour un paramètre Systems Manager.

`domain`

Type : String

Obligatoire : oui

Nom de domaine complet hébergé par un annuaire AWS Directory Service for Microsoft Active Directory(AWS Managed Microsoft AD) ou un Active Directory auto-hébergé EC2 .

Méthodes de stockage des informations d'identification de volume FSx pour le serveur de fichiers Windows

Il existe deux méthodes différentes de stockage des informations d'identification à utiliser avec le paramètre d'informations d'identification.

- AWS Secrets Manager secret

  Ces informations d'identification peuvent être créées dans la AWS Secrets Manager console à l'aide de la catégorie Autre type de secret. Vous ajoutez une ligne pour chaque key/value paire username/admin et un mot de passe/*`password`*.
- Paramètre Systems Manager

  Ces informations d'identification peuvent être créées dans la console de paramètres Systems Manager en saisissant du texte dans le formulaire qui se trouve dans l'exemple d'extrait de code suivant.

```
{
  "username": "admin",
  "password": "password"
}
```

Le paramètre `credentialsParameter` dans la définition de tâche `FSxWindowsFileServerVolumeConfiguration` contient l'ARN de secret ou l'ARN du paramètre Systems Manager. Pour de plus amples informations, veuillez consulter les rubriques Présentation d' AWS Secrets Manager dans le Guide de l'utilisateur de Secrets Manager et Systems Manager Parameter Store dans le Guide de l'utilisateur de Systems Manager.

Découvrez comment configurer FSx les systèmes de fichiers Windows File Server pour Amazon ECS

Découvrez comment lancer une instance Windows optimisée pour Amazon ECS qui héberge un système de fichiers FSx pour Windows File Server et des conteneurs pouvant accéder au système de fichiers. Pour ce faire, vous devez d'abord créer un Microsoft Active Directory AWS Directory Service AWS géré. Ensuite, vous créez un système de fichiers et un cluster FSx pour Windows File Server File Server avec une EC2 instance Amazon et une définition de tâche. Vous configurez la définition de tâche pour vos conteneurs afin qu'ils utilisent le système de fichiers FSx for Windows File Server. Enfin, vous testez le système de fichiers.

Cela prend 20 à 45 minutes chaque fois que vous lancez ou supprimez le système FSx de fichiers Active Directory ou Windows File Server. Préparez-vous à réserver au moins 90 minutes pour terminer le didacticiel ou le compléter en quelques sessions.

Prérequis pour le didacticiel

- Un utilisateur administratif. Consultez Configurer l'utilisation d'Amazon ECS.
- (Facultatif) Une paire de `PEM` clés pour la connexion à votre instance EC2 Windows via un accès RDP. Pour plus d'informations sur la création de paires de clés, consultez la section relative aux paires de EC2 clés Amazon et aux EC2 instances Amazon dans le guide de EC2 l'utilisateur Amazon.
- Un VPC avec au moins un sous-réseau public et un sous-réseau privé et un groupe de sécurité. Vous pouvez utiliser votre VPC par défaut. Vous n'avez pas besoin d'une passerelle ou d'un périphérique NAT. AWS Directory Service ne prend pas en charge la traduction d'adresses réseau (NAT) avec Active Directory. Pour que cela fonctionne, le système de fichiers Active Directory FSx pour Windows File Server, le cluster ECS et l'EC2 instance doivent se trouver dans votre VPC. Pour plus d'informations concernant VPCs Active Directories, voir Création d'un VPC et Conditions préalables à la création d'un AWS Microsoft AD géré.
- Les autorisations IAM ecsInstanceRole et ecsTaskExecution Role sont associées à votre compte. Ces rôles liés à un service permettent aux services d'effectuer des appels d'API et d'accéder aux conteneurs, secrets, répertoires et serveurs de fichiers en votre nom.

Étape 1 : Créer des rôles IAM d'accès

Créez un cluster avec la AWS Management Console.

1. Vérifiez si vous en avez un ecsInstanceRole et comment vous pouvez en créer un si vous n'en avez pas. Rôle IAM d'instance de conteneur Amazon ECS

2. Nous recommandons que les stratégies de rôle soient personnalisées pour les autorisations minimales dans un environnement de production réel. Pour suivre ce didacticiel, vérifiez que la politique AWS gérée suivante est attachée à votreecsInstanceRole. Joignez la stratégie si elle n'est pas déjà attachée.

   - EC2ContainerServiceforEC2Rôle Amazon
   - Amazon SSMManaged InstanceCore
   - Amazon SSMDirectory ServiceAccess

   Pour associer des politiques AWS gérées.

   a. Ouvrez la console IAM.
   b. Dans le panneau de navigation, choisissez Roles (Rôles).
   c. Choisissez un rôle géré par AWS .
   d. Choisissez Autorisations, Attacher des stratégies.
   e. Pour réduire le nombre de stratégies disponibles à attacher, utilisez Filter (Filtrer).
   f. Sélectionnez la stratégie appropriée, puis choisissez Attach policy (Attacher la stratégie).

3. Consultez Rôle IAM d'exécution de tâche Amazon ECS pour vérifier si vous avez un ecsTaskExecution rôle et comment vous pouvez en créer un si vous n'en avez pas.

   Nous recommandons que les stratégies de rôle soient personnalisées pour les autorisations minimales dans un environnement de production réel. Pour suivre ce didacticiel, vérifiez que les politiques AWS gérées suivantes sont associées à votre ecsTaskExecution rôle. Attachez les stratégies si elles ne sont pas déjà attachées. Utilisez la procédure décrite dans la section précédente pour associer les politiques AWS gérées.

   - SecretsManagerReadWrite
   - Amazon FSx ReadOnlyAccess
   - Amazon SSMRead OnlyAccess
   - Amazon ECSTask ExecutionRolePolicy

Étape 2 : Créer un Active Directory (AD) Windows

1. Suivez les étapes décrites dans la section Création de votre Microsoft AD AWS géré dans le Guide d'administration du AWS Directory Service. Utilisez le VPC que vous avez désigné

pour ce didacticiel. À l'étape 3 de la création de votre Microsoft AD AWS géré, enregistrez le nom d'utilisateur et le mot de passe administrateur à utiliser lors de l'étape suivante. Notez également le nom DNS complet du répertoire pour les étapes futures. Vous pouvez effectuer l'étape suivante lors de la création d'Active Directory.

2. Créez un AWS secret du Gestionnaire de Secrets à utiliser dans les étapes suivantes. Pour plus d'informations, voir Commencer à utiliser Secrets Manager dans le Guide de l'utilisateur de AWS Secrets Manager.

   a. Ouvrez la console Secrets Manager.
   b. Cliquez sur Store a new secret (Stocker un nouveau secret).
   c. Sélectionnez Other type of secrets (Autres types de secrets).
   d. Pour Secret key/value (Valeur clé secrète) dans la première ligne, créez une clé **`username`** avec la valeur **`admin`**. Cliquez sur + Add row (+ Ajouter une ligne).
   e. Dans la nouvelle ligne, créez une clé **`password`**. Pour obtenir de la valeur, saisissez le mot de passe que vous avez saisi à l'étape 3 de la section Création de votre annuaire AD AWS géré.
   f. Cliquez sur Next (Suivant).
   g. Fournissez un nom et une description de secret. Cliquez sur Next (Suivant).
   h. Cliquez sur Next (Suivant). Cliquez sur Store (Stocker).
   i. À partir de la page Secrets, cliquez sur le secret que vous venez de créer.
   j. Enregistrez l'ARN du nouveau secret pour l'utiliser dans les étapes suivantes.
   k. Vous pouvez passer à l'étape suivante pendant la création de votre répertoire Active Directory.

Étape 3 : Vérifier et mettre à jour votre groupe de sécurité

Dans cette étape, vous vérifiez et mettez à jour les règles du groupe de sécurité que vous utilisez. Pour cela, vous pouvez utiliser le groupe de sécurité par défaut qui a été créé pour votre VPC.

Vérifiez et mettez à jour le groupe de sécurité.

Vous devez créer ou modifier votre groupe de sécurité pour envoyer des données depuis et vers les ports, comme indiqué dans la section Groupes de sécurité Amazon VPC du guide de l'utilisateur FSx pour les serveurs de fichiers Windows. Pour ce faire, créez la règle entrante du groupe de sécurité affichée dans la première ligne du tableau suivant des règles entrantes. Cette règle autorise le trafic entrant à partir d'interfaces réseau (et de leurs instances associées) affectées au groupe de sécurité.

Toutes les ressources cloud que vous créez se trouvent au sein du même VPC et sont rattachées au même groupe de sécurité. Par conséquent, cette règle permet d'envoyer du trafic vers et FSx depuis le système de fichiers Windows File Server, Active Directory et l'instance ECS selon les besoins. Les autres règles entrantes autorisent le trafic à servir le site web et l'accès RDP pour la connexion à votre instance ECS.

Le tableau suivant indique les règles entrantes de groupe de sécurité requises pour ce didacticiel.

Type	Protocole	Plage de ports	Source
Tout le trafic	Tous	Tous	`sg-securitygroup`
HTTPS	TCP	443	0.0.0.0/0
RDP	TCP	3389	l'adresse IP de votre ordinateur portable

Le tableau suivant indique les règles sortantes du groupe de sécurité requises pour ce didacticiel.

Type	Protocole	Plage de ports	Destination
Tout le trafic	Tous	Tous	0.0.0.0/0

1. Ouvrez la EC2 console et sélectionnez Security Groups dans le menu de gauche.
2. Dans la liste des groupes de sécurité qui s'affiche, cochez la case située à gauche du groupe de sécurité que vous utilisez pour ce didacticiel.

   Les informations de votre groupe de sécurité s'affichent.
3. Modifiez les règles entrantes et sortantes en sélectionnant l'onglet Inbound rules (Règles entrantes) ou Outbound rules (Règles sortantes) et en cliquant sur Edit inbound rules (Modifier les règles entrantes) ou Edit outbound rules (Modifier les règles sortantes). Modifiez les règles pour qu'elles correspondent à celles affichées dans les tableaux précédents. Après avoir créé

votre EC2 instance ultérieurement dans ce didacticiel, modifiez la source RDP de la règle entrante avec l'adresse IP publique de votre EC2 instance, comme décrit dans Connect to your Windows instance using RDP du Amazon EC2 User Guide.

Étape 4 : Création d'un système de fichiers FSx pour Windows File Server

Une fois que votre groupe de sécurité a été vérifié et mis à jour et que votre Active Directory a été créé et est en état actif, créez le système de fichiers FSx pour Windows File Server dans le même VPC que votre Active Directory. Suivez les étapes ci-dessous pour créer un système de fichiers FSx pour serveur de fichiers Windows pour vos tâches Windows.

Créez votre premier système de fichiers.

1. Ouvrez la FSx console Amazon.
2. Dans Sur le tableau de bord, choisissez Create file system (Créer un système de fichiers) pour ouvrir l'assistant de création de système de fichiers.
3. Sur la page Sélectionner le type de système FSx de fichiers, choisissez Windows File Server, puis Next. La page Create file system (Créer un système de fichiers) s'affiche.
4. Dans la section File system details (Informations du système de fichiers), indiquez un nom pour votre système de fichiers. Donner un nom à vos systèmes de fichiers facilite leur recherche et leur gestion. Vous pouvez utiliser jusqu'à 256 caractères Unicode. Les caractères autorisés sont les lettres, les chiffres, les espaces et les caractères spéciaux suivants : signe plus (+), signe moins (-), signe égal (=), point (.), trait de soulignement (_), deux points (:) et barre oblique (/).
5. Pour Deployment type (Type de déploiement), choisissez Single-AZ (Mono-AZ) afin de déployer un système de fichiers qui est déployé dans une seule zone de disponibilité. Mono-AZ 2 est la dernière génération de systèmes de fichiers de zone de disponibilité unique et prend en charge le stockage SSD et HDD.
6. Pour Storage type (Type de stockage), choisissez HDD.
7. Pour Storage capacity (Capacité de stockage), saisissez la capacité de stockage minimale.
8. Conservez la valeur par défaut de Throughput capacity (Capacité de débit).
9. Dans la section Réseau et sécurité, choisissez le même Amazon VPC que celui que vous avez choisi pour votre AWS Directory Service annuaire.
10. Pour VPC Security Groups (Groupes de sécurité VPC), choisissez le groupe de sécurité que vous avez vérifié à l'Étape 3 : Vérifier et mettre à jour votre groupe de sécurité.

11. Pour Windows authentication (Authentification Windows), choisissez AWS Managed Microsoft Active Directory (Annuaire Microsoft Active Directory géré par), puis choisissez votre répertoire AWS Directory Service dans la liste.
12. Pour Encryption (Chiffrement), conservez la valeur par défaut du paramètre Encryption key (Clé de chiffrement), aws/fsx (default) (aws/fsx [par défaut]).
13. Conservez les paramètres par défaut pour Maintenance preferences (Préférences de maintenance).
14. Cliquez sur Next (Suivant).
15. Vérifiez la configuration du système de fichiers qui s'affiche sur la page Create file system (Créer un système de fichiers). Pour référence, notez les paramètres du système de fichiers que vous pouvez modifier une fois le système de fichiers créé. Choisissez Create file system (Créer un système de fichiers).
16. Notez l'ID du système de fichiers. Vous en aurez besoin dans une étape ultérieure.

    Vous pouvez passer aux étapes suivantes pour créer un cluster et une EC2 instance lors de la création du système de fichiers FSx for Windows File Server.

Étape 5 : créer un cluster Amazon ECS

Création d'un cluster à l'aide de la console Amazon ECS

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Cluster configuration (Configuration de cluster), pour Cluster name (Nom du cluster), saisissez windows-fsx-cluster.
6. Développez l'infrastructure, AWS Fargate effacez (sans serveur), puis sélectionnez les EC2 instances Amazon.
   - Pour créer un groupe Auto Scaling, depuis Auto Scaling group (ASG) (Groupe Auto Scaling [ASG]), sélectionnez Create new group (Créer un nouveau groupe), puis fournissez les détails suivants sur le groupe :
     - Dans Système d'exploitation/Architecture, choisissez Windows Server 2019 Core.

- EC2 Par type d'instance, choisissez t2.medium ou t2.micro.

7. Choisissez Créer.

Étape 6 : créer une EC2 instance Amazon optimisée pour Amazon ECS

Créez une instance de conteneur Windows Amazon ECS.

Pour créer une instance Amazon ECS

1. Utilisez la commande `aws ssm get-parameters` pour récupérer le nom d'AMI de la région qui héberge votre VPC. Pour plus d'informations, consultez Extraction des métadonnées d'AMI optimisée pour Amazon ECS.
2. Utilisez la EC2 console Amazon pour lancer l'instance.
   a. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
   b. Dans la barre de navigation, sélectionnez la région à utiliser.
   c. Dans le EC2 tableau de bord, choisissez Launch instance.
   d. Pour Name (Nom), saisissez un nom unique.
   e. Pour Images de l'application et du SE (Amazon Machine Image), dans le champ de recherche, saisissez le nom de l'AMI que vous avez récupérée.
   f. Pour Type d'instance, choisissez t2.medium ou t2.micro.
   g. Pour Key pair (login) (Paire de clés (connexion)), choisissez une paire de clés. Si vous ne spécifiez pas de paire de clés, vous
   h. Sous Paramètres réseau, pour VPC et Sous-réseau, choisissez votre VPC et un sous-réseau public.
   i. Sous Network Settings (Paramètres réseau), pour Security group (Groupe de sécurité), choisissez un groupe de sécurité existant ou créez-en un nouveau. Assurez-vous que le groupe de sécurité que vous choisissez possède les règles entrantes et sortantes définies dans Prérequis pour le didacticiel.
   j. Sous Network settings (Paramètres réseau), pour Auto-assign Public IP (Attribuer automatiquement l'adresse IP publique), sélectionnez Enable (Activer).
   k. Développez Détails avancés, puis pour Répertoire de jonction de domaines, sélectionnez l'ID de l'Active Directory que vous avez créé. Ce domaine optionnel rejoint votre AD lorsque l' EC2 instance est lancée.
   l. Sous Détails avancés, pour le profil d'instance IAM, sélectionnez ecsInstanceRole.

m. Configurez votre instance de conteneur Amazon ECS avec les données utilisateur suivantes. Sous Détails avancés, collez le script suivant dans le champ Données utilisateur, en le *cluster_name* remplaçant par le nom de votre cluster.

```
<powershell>
Initialize-ECSAgent -Cluster windows-fsx-cluster -EnableTaskIAMRole
</powershell>
```

n. Lorsque vous êtes prêt, cochez la case de confirmation, puis sélectionnez Launch Instances (Lancer des instances).

o. Une page de confirmation indique que l'instance est en cours de lancement. Sélectionnez View Instances (Afficher les instances) pour fermer la page de confirmation et revenir à la console.

3. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
4. Dans le volet de navigation, choisissez Clusters, puis choisissez windows-fsx-cluster.
5. Choisissez l'onglet Infrastructure et vérifiez que votre instance a été enregistrée dans le windows-fsx-clustercluster.

Étape 7 : Enregistrement d'une définition de tâche Windows

Avant de pouvoir exécuter des conteneurs Windows dans votre cluster Amazon ECS, vous devez enregistrer une définition de tâche. L'exemple de définition de tâche suivant affiche une page Web simple. La tâche lance deux conteneurs qui ont accès au système de FSx fichiers. Le premier conteneur écrit un fichier HTML dans le système de fichiers. Le deuxième conteneur télécharge le fichier HTML à partir du système de fichiers et sert la page web.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
4. Dans la zone de l'éditeur JSON, remplacez les valeurs de votre rôle d'exécution des tâches et les informations relatives à votre système de FSx fichiers, puis choisissez Enregistrer.

```
{
    "containerDefinitions": [
        {
            "entryPoint": [
```

```
                "powershell",
                "-Command"
            ],
            "portMappings": [],
            "command": ["New-Item -Path C:\\fsx-windows-dir\\index.html -ItemType
 file -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body
 {margin-top: 40px; background-color: #333;} </style> </head><body> <div
 style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1> <h2>It
 Works!</h2> <p>You are using Amazon FSx for Windows File Server file system for
 persistent container storage.</p>' -Force"],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": false,
            "name": "container1",
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ]
        },
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [
                {
                    "hostPort": 443,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": ["Remove-Item -Recurse C:\\inetpub\\wwwroot\\* -Force;
 Start-Sleep -Seconds 120; Move-Item -Path C:\\fsx-windows-dir\\index.html -
Destination C:\\inetpub\\wwwroot\\index.html -Force; C:\\ServiceMonitor.exe
 w3svc"],
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
```

```
                    "readOnly": false
                }
            ],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": true,
            "name": "container2"
        }
    ],
    "family": "fsx-windows",
    "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
    "volumes": [
        {
            "name": "fsx-windows-dir",
            "fsxWindowsFileServerVolumeConfiguration": {
                "fileSystemId": "fs-0eeb5730b2EXAMPLE",
                "authorizationConfig": {
                    "domain": "example.com",
                    "credentialsParameter": "arn:arn-1234"
                },
                "rootDirectory": "share"
            }
        }
    ]
}
```

Étape 8 : Exécuter une tâche et afficher les résultats

Avant d'exécuter la tâche, vérifiez que l'état de votre système de fichiers FSx pour Windows File Server est disponible. Si c'est le cas, vous pouvez exécuter une tâche à l'aide de la définition de tâche que vous avez créée. La tâche commence par créer des conteneurs qui remanient un fichier HTML sur eux à l'aide du système de fichiers. Après le remaniement, un serveur web sert la page HTML simple.

> **Note**
>
> Il se peut que vous ne parveniez pas à vous connecter au site web depuis un VPN.

Exécutez une tâche et affichez les résultats à l'aide de la console Amazon ECS.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le volet de navigation, choisissez Clusters, puis choisissez windows-fsx-cluster.
3. Choisissez l'onglet Tâches, puis Exécuter une nouvelle tâche.
4. Pour Type de lancement, choisissez EC2.
5. Sous Configuration du déploiement, pour Définition de tâche, sélectionnez fsx-windows, puis sélectionnez Créer.
6. Lorsque le statut de votre tâche est EN COURS D'EXÉCUTION, sélectionnez l'ID de la tâche.
7. Sous Conteneurs, lorsque le statut container1 est ARRÊTÉ, sélectionnez container2 pour afficher les détails du conteneur.
8. Dans Informations sur le conteneur pour container2, sélectionnez Liaisons réseau, puis cliquez sur l'adresse IP externe associée au conteneur. Votre navigateur s'ouvre et affiche le message suivant.

```
Amazon ECS Sample App
It Works!
You are using Amazon FSx for Windows File Server file system for persistent
 container storage.
```

> **Note**
>
> L'affichage du message peut prendre quelques minutes. Si ce message ne s'affiche pas après plusieurs minutes, vérifiez que vous n'exécutez pas dans un VPN et assurez-vous que le groupe de sécurité de votre instance de conteneur autorise le trafic HTTP réseau entrant sur le port 443.

Étape 9 : nettoyer

> **Note**
>
> La suppression du système de fichiers Windows File Server ou FSx de l'AD prend 20 à 45 minutes. Vous devez attendre que les opérations FSx de suppression du système de fichiers du serveur de fichiers Windows soient terminées avant de démarrer les opérations de suppression AD.

Supprimer FSx pour le système de fichiers Windows File Server.

1. Ouvrez la FSx console Amazon
2. Cliquez sur le bouton radio situé à gauche du système FSx de fichiers Windows File Server que vous venez de créer.
3. Choisissez Actions.
4. Sélectionnez Delete file system (Supprimer le système de fichiers).

Supprimez le répertoire AD.

1. Ouvrez la AWS Directory Service console.
2. Cliquez sur la case d'option située à gauche du répertoire AD que vous venez de créer.
3. Choisissez Actions.
4. Sélectionnez Delete directory (Supprimer le répertoire).

Supprimez le cluster.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le volet de navigation, choisissez Clusters, puis choisissez windows-fsx-cluster.
3. Choisissez Supprimer le cluster.
4. Saisissez l'expression, puis choisissez Supprimer.

Mettre fin à EC2 l'instance.

1. Ouvrez la EC2 console Amazon.
2. Dans le menu de gauche, sélectionnez Instances.
3. Cochez la case située à gauche de l' EC2 instance que vous avez créée.
4. Cliquez sur État de l'instance, puis sur Résilier l'instance.

Supprimez le secret.

1. Ouvrez la console Secrets Manager.
2. Sélectionnez le secret que vous avez créé pour cette démonstration.

3. Cliquez sur Actions.
4. Sélectionnez Delete secret (Supprimer le secret).

Utiliser des volumes Docker avec Amazon ECS

Lorsque vous utilisez des volumes Docker, le pilote `local` intégré ou un pilote de volume tiers peut être utilisé. Les volumes Docker sont gérés par Docker et un répertoire est créé dans `/var/lib/docker/volumes` sur l'instance de conteneur qui contient les données du volume.

Pour utiliser des volumes Docker, spécifiez un `dockerVolumeConfiguration` dans votre définition de tâche. Pour plus d'informations, consultez la section Volumes dans la Docker documentation.

Certains cas d'utilisation courants pour les volumes Docker sont les suivants :

- Pour fournir des volumes de données permanent à utiliser avec les conteneurs
- Pour partager un volume de données défini à différents emplacements sur différents conteneurs situés sur la même instance de conteneur
- Pour définir un volume de données vide, non permanent et le monter dans plusieurs conteneurs au sein d'une même tâche
- Pour fournir un volume de données à votre tâche qui est gérée par un pilote tiers

Considérations relatives à l'utilisation des volumes Docker

Tenez compte des éléments suivants lorsque vous utilisez des volumes Docker :

- Les volumes Docker ne sont pris en charge que lors de l'utilisation du type de EC2 lancement ou d'instances externes.
- Les conteneurs Windows prennent uniquement en charge l'utilisation du pilote `local`.
- Si un pilote tiers est utilisé, assurez-vous qu'il est installé et actif sur l'instance de conteneur avant le démarrage de l'agent de conteneur. Si le pilote tiers n'est pas actif avant le démarrage de l'agent, vous pouvez redémarrer l'agent de conteneur à l'aide de l'une des commandes suivantes :
  - Pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS :

    ```
    sudo systemctl restart ecs
    ```

  - Pour l'AMI Amazon Linux optimisée pour Amazon ECS :

```
sudo stop ecs && sudo start ecs
```

Pour plus d'informations sur la manière de spécifier un volume Docker dans une définition de tâche, consultezSpécifier un volume Docker dans une définition de tâche Amazon ECS.

Spécifier un volume Docker dans une définition de tâche Amazon ECS

Avant que vos conteneurs puissent utiliser des volumes de données, vous devez spécifier les configurations du volume et du point de montage dans votre définition de tâche. Cette section décrit la configuration du volume pour un conteneur. Pour les tâches qui utilisent un volume Docker, spécifiez une propriété `dockerVolumeConfiguration`. Pour les tâches qui utilisent un volume hôte de montage lié, spécifiez un `host` et éventuellement un `sourcePath`.

La définition de tâche JSON indiquée ci-dessous illustre la syntaxe des objets `volumes` et `mountPoints` pour un conteneur :

```
{
    "containerDefinitions": [
        {
            "mountPoints": [
                {
                    "sourceVolume": "string",
                    "containerPath": "/path/to/mount_volume",
                    "readOnly": boolean
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "string",
            "dockerVolumeConfiguration": {
                "scope": "string",
                "autoprovision": boolean,
                "driver": "string",
                "driverOpts": {
                    "key": "value"
                },
                "labels": {
                    "key": "value"
```

```
                }
            }
        }
    ]
}
```

`name`

Type : chaîne

Obligatoire : non

Nom du volume. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, tirets () et traits de soulignement (`-`) sont autorisés. _ Ce nom est référencé dans le `sourceVolume` paramètre de l'`mountPoints`objet de définition du conteneur.

`dockerVolumeConfiguration`

Type : DockerVolumeConfigurationObjet

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Docker. Les volumes Docker ne sont pris en charge que lors de l'exécution de tâches sur des EC2 instances. Les conteneurs Windows ne prennent en charge que l'utilisation du `local` pilote. Pour utiliser des montages liés, spécifiez plutôt un paramètre `host`.

`scope`

Type : String

Valeurs valides : `task` | `shared`

Obligatoire : non

Portée du volume Docker, qui détermine son cycle de vie. Les volumes Docker destinés à un élément `task` sont automatiquement mis en service lorsque la tâche commence, et détruits lorsque la tâche s'arrête. Les volumes Docker définis comme `shared` ne sont pas supprimés lorsque la tâche s'arrête.

`autoprovision`

Type : booléen

Valeur par défaut : `false`

Obligatoire : non

Si cette valeur est `true`, le volume Docker est créé s'il n'existe pas déjà. Ce champ n'est utilisé que si `scope` c'est le cas`shared`. Si tel `scope` est le cas`task`, ce paramètre doit être omis.

`driver`

Type : chaîne

Obligatoire : non

Pilote de volume Docker à utiliser. La valeur du pilote doit correspondre au nom du pilote fourni par Docker car ce nom est utilisé pour le placement des tâches. Si le pilote a été installé à l'aide de la CLI du plugin Docker, utilisez-le `docker plugin ls` pour récupérer le nom du pilote depuis votre instance de conteneur. Si le pilote a été installé à l'aide d'une autre méthode, utilisez Docker plugin discovery pour récupérer le nom du pilote.

`driverOpts`

Type : chaîne

Obligatoire : non

Une carte des options spécifiques au pilote Docker à utiliser. Ce paramètre correspond `DriverOpts` à la section Créer un volume de Docker.

`labels`

Type : chaîne

Obligatoire : non

Métadonnées personnalisées à ajouter à votre volume Docker.

`mountPoints`

Type : tableau d'objets

Obligatoire : non

Les points de montage des volumes de données de votre conteneur. Ce paramètre correspond `Volumes` à l'API Docker create-container et `--volume` à l'option docker run.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`. Les conteneurs Windows ne peuvent pas monter de répertoires sur un

autre lecteur, et les points de montage ne peuvent pas être utilisés sur plusieurs lecteurs. Vous devez spécifier des points de montage pour associer un volume Amazon EBS directement à une tâche Amazon ECS.

`sourceVolume`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Le chemin dans le conteneur où le volume sera monté.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

Pour les tâches exécutées sur EC2 des instances exécutant le système d'exploitation Windows, laissez la valeur par défaut de`false`.

Exemples de volumes Docker pour Amazon ECS

Les exemples suivants montrent comment fournir un stockage éphémère pour un conteneur et comment fournir un volume partagé pour plusieurs conteneurs, et comment fournir un stockage persistant NFS pour un conteneur.

Pour fournir un stockage éphémère à un conteneur à l'aide d'un volume Docker

Dans cet exemple, un conteneur utilise un volume de données vide qui est éliminé une fois la tâche terminée. Un exemple d'utilisation est que vous pourriez avoir un conteneur qui doit accéder à un emplacement de stockage de fichiers de travail pendant une tâche. Cette tâche peut être réalisée à l'aide d'un volume Docker.

1. Dans la section `volumes` de la définition de tâche, définissez un volume de données avec les valeurs `name` et `DockerVolumeConfiguration`. Dans cet exemple, nous spécifions la portée sous la forme `task`. Le volume est donc supprimé après l'arrêt de la tâche et il utilise le pilote `local` intégré.

```
"volumes": [
    {
        "name": "scratch",
        "dockerVolumeConfiguration" : {
            "scope": "task",
            "driver": "local",
            "labels": {
                "scratch": "space"
            }
        }
    }
]
```

2. Dans la section `containerDefinitions`, définissez un conteneur avec des valeurs `mountPoints` qui font référence au nom du volume défini et la valeur `containerPath` sur laquelle monter le volume sur le conteneur.

```
"containerDefinitions": [
    {
        "name": "container-1",
        "mountPoints": [
            {
              "sourceVolume": "scratch",
              "containerPath": "/var/scratch"
            }
        ]
    }
]
```

Pour fournir un stockage persistant à plusieurs conteneurs à l'aide d'un volume Docker

Dans cet exemple, vous voulez un volume partagé qui sera utilisé par plusieurs conteneurs et vous souhaitez qu'il persiste après l'arrêt des tâches qui l'utilisent. Le pilote `local` intégré est en cours d'utilisation. Ainsi, le volume est toujours lié au cycle de vie de l'instance de conteneur.

1. Dans la section `volumes` de la définition de tâche, définissez un volume de données avec les valeurs `name` et `DockerVolumeConfiguration`. Dans cet exemple, spécifiez une portée `shared` pour que le volume persiste, définissez autoprovision sur `true`. C'est ainsi que le volume est créé pour être utilisé. Ensuite, utilisez également le pilote `local` intégré.

```
"volumes": [
    {
        "name": "database",
        "dockerVolumeConfiguration" : {
            "scope": "shared",
            "autoprovision": true,
            "driver": "local",
            "labels": {
                "database": "database_name"
            }
        }
    }
]
```

2. Dans la section `containerDefinitions`, définissez un conteneur avec des valeurs `mountPoints` qui font référence au nom du volume défini et la valeur `containerPath` sur laquelle monter le volume sur le conteneur.

```
"containerDefinitions": [
    {
        "name": "container-1",
        "mountPoints": [
        {
          "sourceVolume": "database",
          "containerPath": "/var/database"
        }
      ]
    },
    {
      "name": "container-2",
      "mountPoints": [
        {
          "sourceVolume": "database",
          "containerPath": "/var/database"
        }
      ]
    }
```

```
  ]
```

Pour fournir un stockage permanent NFS pour un conteneur à l'aide d'un volume Docker

Dans cet exemple, un conteneur utilise un volume de données NFS qui est automatiquement monté lorsque la tâche démarre et démonté lorsque la tâche s'arrête. Cela utilise le pilote `local` intégré Docker. Un exemple d'utilisation est que vous pourriez avoir un stockage NFS local et que vous devez y accéder à l'aide d'une tâche ECS Anywhere. Cela peut être réalisé à l'aide d'un volume Docker avec option de pilote NFS.

1. Dans la section `volumes` de la définition de tâche, définissez un volume de données avec les valeurs `name` et `DockerVolumeConfiguration`. Dans cet exemple, spécifiez une portée `task` de telle sorte que le volume soit démonté une fois la tâche terminée. Utilisez le pilote `local` et configurez le `driverOpts` avec le `type`, `device` et les options o en conséquence. Remplacez `NFS_SERVER` par le point de terminaison du serveur NFS.

```
"volumes": [
       {
           "name": "NFS",
           "dockerVolumeConfiguration" : {
               "scope": "task",
               "driver": "local",
               "driverOpts": {
                   "type": "nfs",
                   "device": "$NFS_SERVER:/mnt/nfs",
                   "o": "addr=$NFS_SERVER"
               }
           }
       }
   ]
```

2. Dans la section `containerDefinitions`, définissez un conteneur avec des valeurs `mountPoints` qui font référence au nom du volume défini et la valeur `containerPath` sur laquelle monter le volume sur le conteneur.

```
"containerDefinitions": [
       {
           "name": "container-1",
           "mountPoints": [
               {
```

```
                "sourceVolume": "NFS",
                "containerPath": "/var/nfsmount"
            }
        ]
    }
]
```

Utiliser des montages par liaison avec Amazon ECS

Avec les montages par liaison, un fichier ou un répertoire sur un hôte, tel qu'une EC2 instance Amazon, est monté dans un conteneur. Les montages par liaison sont pris en charge pour les tâches hébergées à la fois sur les instances Fargate et Amazon. EC2 Les montages de liaison sont liés au cycle de vie du conteneur qui les utilise. Une fois que tous les conteneurs qui utilisent un montage lié sont arrêtés, par exemple lorsqu'une tâche est arrêtée, les données sont supprimées. Pour les tâches hébergées sur des EC2 instances Amazon, les données peuvent être liées au cycle de vie de l' EC2 instance Amazon hôte en spécifiant une `host` `sourcePath` valeur facultative dans votre définition de tâche. Pour plus d'informations, consultez la section Bind mounts dans la documentation Docker.

Voici quelques cas d'utilisation courants pour les montages liés.

- Pour fournir un volume de données vide à monter dans un ou plusieurs conteneurs.
- Pour monter un volume de données hôte dans un ou plusieurs conteneurs.
- Pour partager un volume de données d'un conteneur source avec d'autres conteneurs dans la même tâche.
- Pour exposer un chemin d'accès et son contenu d'un fichier Dockerfile à un ou plusieurs conteneurs.

Considérations relatives à l'utilisation des montages liés

Lorsque vous utilisez des montages liés, tenez compte des éléments suivants.

- Par défaut, les tâches hébergées sur AWS Fargate une version de plate-forme `1.4.0` ou ultérieure (Linux) `1.0.0` ou ultérieure (Windows) reçoivent un minimum de 20 GiB de stockage éphémère pour les montages liés. Vous pouvez augmenter la quantité totale de stockage éphémère jusqu'à un maximum de 200 GiB en spécifiant le `ephemeralStorage` paramètre dans la définition de votre tâche.

- Pour exposer des fichiers d'un fichier Dockerfile à un volume de données lorsqu'une tâche est exécutée, le plan de données Amazon ECS recherche une directive `VOLUME`. Si le chemin absolu spécifié dans la directive `VOLUME` est le même que le `containerPath` spécifié dans la définition de tâche, les données du chemin de directive `VOLUME` sont copiées sur le volume de données. Dans l'exemple Dockerfile suivant, un fichier nommé `examplefile` dans le répertoire `/var/log/exported` est écrit sur l'hôte, puis monté à l'intérieur du conteneur.

  ```
  FROM public.ecr.aws/amazonlinux/amazonlinux:latest
  RUN mkdir -p /var/log/exported
  RUN touch /var/log/exported/examplefile
  VOLUME ["/var/log/exported"]
  ```

  Par défaut, les autorisations de volume sont définies sur `0755` et le propriétaire en tant que `root`. Vous pouvez personnaliser ces autorisations dans le fichier Dockerfile. L'exemple suivant définit le propriétaire du répertoire en tant que `node`.

  ```
  FROM public.ecr.aws/amazonlinux/amazonlinux:latest
  RUN yum install -y shadow-utils && yum clean all
  RUN useradd node
  RUN mkdir -p /var/log/exported && chown node:node /var/log/exported
  RUN touch /var/log/exported/examplefile
  USER node
  VOLUME ["/var/log/exported"]
  ```

- Pour les tâches hébergées sur des EC2 instances Amazon, lorsqu'aucune `sourcePath` valeur `host` et ne sont spécifiées, le daemon Docker gère le montage de liaison pour vous. Lorsqu'aucun conteneur ne fait référence à ce montage lié, le service de nettoyage des tâches de l'agent de conteneur Amazon ECS finit par le supprimer. Par défaut, cela se produit trois heures après la sortie du conteneur. Toutefois, vous pouvez configurer cette durée avec la variable de l'agent `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS. Si vous avez besoin de conserver ces données au-delà du cycle de vie de conteneur, spécifiez une valeur `sourcePath` pour le montage lié.

Spécifier un montage de liaison dans une définition de tâche Amazon ECS

Pour les tâches Amazon ECS hébergées sur des instances Fargate ou EC2 Amazon, l'extrait de code JSON de définition de tâche suivant indique la syntaxe des objets`ephemeralStorage`, et `volumes` pour une définition `mountPoints` de tâche.

```
{
   "family": "",
   ...
   "containerDefinitions" : [
      {
         "mountPoints" : [
            {
               "containerPath" : "/path/to/mount_volume",
               "sourceVolume" : "string"
            }
          ],
          "name" : "string"
       }
    ],
    ...
    "volumes" : [
       {
          "name" : "string"
       }
    ],
    "ephemeralStorage": {
    "sizeInGiB": integer
    }
}
```

Pour les tâches Amazon ECS hébergées sur des EC2 instances Amazon, vous pouvez utiliser le `host` paramètre facultatif et un `sourcePath` lorsque vous spécifiez les détails du volume de tâches. Lorsqu'il est spécifié, il lie le montage lié au cycle de vie de la tâche plutôt qu'au conteneur.

```
"volumes" : [
    {
        "host" : {
            "sourcePath" : "string"
        },
        "name" : "string"
    }
]
```

Voici des descriptions plus détaillées de chaque paramètre de définition de tâche.

`name`

Type : chaîne

Obligatoire : non

Nom du volume. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, tirets () et traits de soulignement (`-`) sont autorisés. _ Ce nom est référencé dans le `sourceVolume` paramètre de l'`mountPoints`objet de définition du conteneur.

`host`

Obligatoire : non

Le `host` paramètre est utilisé pour lier le cycle de vie du montage de liaison à l' EC2 instance Amazon hôte, plutôt qu'à la tâche et à l'endroit où il est stocké. Si le paramètre `host` est vide, le démon Docker attribue un chemin hôte au volume de données, mais la persistance des données après l'arrêt des conteneurs qui lui sont associés n'est pas garantie.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`.

> **Note**
>
> Le `sourcePath` paramètre n'est pris en charge que lors de l'utilisation de tâches hébergées sur EC2 des instances Amazon.

`sourcePath`

Type : chaîne

Obligatoire : non

Lorsque le `host` paramètre est utilisé, spécifiez a `sourcePath` pour déclarer le chemin sur l' EC2 instance Amazon hôte qui est présentée au conteneur. Si ce paramètre est vide, le démon Docker attribue un chemin hôte pour vous. Si le `host` paramètre contient un emplacement de `sourcePath` fichier, le volume de données est conservé à l'emplacement spécifié sur l' EC2 instance Amazon hôte jusqu'à ce que vous le supprimiez manuellement. Si la `sourcePath` valeur n'existe pas sur l' EC2 instance Amazon hôte, le démon Docker la crée. Si l'emplacement n'existe pas, le contenu du chemin source est exporté.

`mountPoints`

Type : tableau d'objets

Obligatoire : non

Les points de montage des volumes de données de votre conteneur. Ce paramètre correspond `Volumes` à l'API Docker create-container et `--volume` à l'option docker run.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`. Les conteneurs Windows ne peuvent pas monter de répertoires sur un autre lecteur, et les points de montage ne peuvent pas être utilisés sur plusieurs lecteurs. Vous devez spécifier des points de montage pour associer un volume Amazon EBS directement à une tâche Amazon ECS.

`sourceVolume`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Le chemin dans le conteneur où le volume sera monté.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

Pour les tâches exécutées sur EC2 des instances exécutant le système d'exploitation Windows, laissez la valeur par défaut de`false`.

`ephemeralStorage`

Type : objet

Obligatoire : non

Quantité de magasin éphémère à allouer pour la tâche. Ce paramètre est utilisé pour augmenter la quantité totale de stockage éphémère disponible, au-delà de la quantité par défaut, pour les tâches hébergées sur AWS Fargate une version de plate-forme `1.4.0` ou ultérieure (Linux) `1.0.0` ou ultérieure (Windows).

Vous pouvez utiliser la CLI Copilot CloudFormation, le AWS SDK ou l'interface de ligne de commande pour spécifier un stockage éphémère pour un montage par liaison.

Exemples de montage par liaison pour Amazon ECS

Les exemples suivants présentent les cas d'utilisation courants liés à l'utilisation d'un support de liaison pour vos conteneurs.

Pour allouer une quantité accrue d'espace de stockage éphémère pour une tâche Fargate

Pour les tâches Amazon ECS hébergées sur Fargate à l'aide de la version `1.4.0` de la plateforme ou une version ultérieure (Linux) ou `1.0.0` (Windows), vous pouvez allouer plus que la quantité de magasins éphémères par défaut pour les conteneurs de votre tâche à utiliser. Cet exemple peut être intégré dans les autres exemples afin d'allouer plus de stockage éphémère pour vos tâches Fargate.

- Dans la définition de la tâche, définissez un objet `ephemeralStorage`. La valeur `sizeInGiB` doit être un nombre entier compris entre `21` et `200` est exprimée en GiB.

```
"ephemeralStorage": {
    "sizeInGiB": integer
}
```

Pour fournir un volume de données vide pour un ou plusieurs conteneurs

Dans certains cas, vous voudrez fournir aux conteneurs dans une tâche un peu d'espace vide. Supposons par exemple que deux conteneurs de base de données doivent accéder au même emplacement de stockage temporaire pendant une tâche. Cela peut être réalisé à l'aide d'un montage lié.

1. Dans la section `volumes` de la définition de tâche, définissez un montage lié nommé `database_scratch`.

```
  "volumes": [
    {
      "name": "database_scratch"
    }
  ]
```

2. Dans la section `containerDefinitions`, créez les définitions de conteneur de base de données. Cela permet de monter le volume.

```
"containerDefinitions": [
    {
      "name": "database1",
      "image": "my-repo/database",
      "cpu": 100,
      "memory": 100,
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "database_scratch",
          "containerPath": "/var/scratch"
        }
      ]
    },
    {
      "name": "database2",
      "image": "my-repo/database",
      "cpu": 100,
      "memory": 100,
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "database_scratch",
          "containerPath": "/var/scratch"
        }
      ]
    }
  ]
```

Pour exposer un chemin d'accès et son contenu d'un fichier Dockerfile à un ou plusieurs conteneurs.

Dans cet exemple, vous avez un fichier Dockerfile qui écrit les données que vous souhaitez monter à l'intérieur d'un conteneur. Cet exemple fonctionne pour les tâches hébergées sur des instances Fargate ou EC2 Amazon.

1. Créez un fichier Dockerfile. L'exemple suivant utilise l'image du conteneur Amazon Linux 2 publique et crée un fichier nommé `examplefile` dans le répertoire `/var/log/exported` que nous voulons monter à l'intérieur du conteneur. Le répertoire `VOLUME` doit spécifier un chemin absolu.

   ```
   FROM public.ecr.aws/amazonlinux/amazonlinux:latest
   RUN mkdir -p /var/log/exported
   RUN touch /var/log/exported/examplefile
   VOLUME ["/var/log/exported"]
   ```

   Par défaut, les autorisations de volume sont définies sur `0755` et le propriétaire en tant que `root`. Ces autorisations peuvent être modifiées dans le fichier Dockerfile. L'exemple suivant définit le propriétaire du répertoire `/var/log/exported` en tant que `node`.

   ```
   FROM public.ecr.aws/amazonlinux/amazonlinux:latest
   RUN yum install -y shadow-utils && yum clean all
   RUN useradd node
   RUN mkdir -p /var/log/exported && chown node:node /var/log/exported
   USER node
   RUN touch /var/log/exported/examplefile
   VOLUME ["/var/log/exported"]
   ```

2. Dans la section `volumes` de la définition de tâche, définissez un volume nommé `application_logs`.

   ```
   "volumes": [
     {
       "name": "application_logs"
     }
   ]
   ```

3. Dans la section `containerDefinitions`, créez les définitions de conteneur d'application. Cela permet de monter le stockage. La valeur `containerPath` doit correspondre au chemin absolu spécifié dans la directive `VOLUME` du Dockerfile.

```
"containerDefinitions": [
  {
    "name": "application1",
    "image": "my-repo/application",
    "cpu": 100,
    "memory": 100,
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "application_logs",
        "containerPath": "/var/log/exported"
      }
    ]
  },
  {
    "name": "application2",
    "image": "my-repo/application",
    "cpu": 100,
    "memory": 100,
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "application_logs",
        "containerPath": "/var/log/exported"
      }
    ]
  }
]
```

Pour fournir un volume de données vide pour un conteneur lié au cycle de vie de l' EC2 instance Amazon hôte

Pour les tâches hébergées sur des EC2 instances Amazon, vous pouvez utiliser des montages par liaison et associer les données au cycle de vie de l' EC2 instance Amazon hôte. Vous pouvez effectuer cette opération en utilisant le paramètre `host` et en spécifiant une valeur `sourcePath`. Tous les fichiers qui existent sur le `sourcePath` sont présentés dans les conteneurs à la valeur `containerPath`. Tous les fichiers écrits dans la `containerPath` valeur sont écrits dans la `sourcePath` valeur de l' EC2 instance Amazon hôte.

> **Important**
>
> Amazon ECS ne synchronise pas votre stockage entre les EC2 instances Amazon. Les tâches qui utilisent le stockage persistant peuvent être placées sur n'importe quelle EC2 instance Amazon de votre cluster disposant d'une capacité disponible. Si vos tâches nécessitent un stockage permanent après l'arrêt et le redémarrage, spécifiez toujours la même EC2 instance Amazon au moment du lancement de la tâche à l'aide de la commande AWS CLI start-task. Vous pouvez également utiliser des volumes Amazon EFS pour le stockage permanent. Pour de plus amples informations, veuillez consulter Utiliser les volumes Amazon EFS avec Amazon ECS.

1. Dans la section `volumes` de la définition de tâche, définissez un montage lié avec les valeurs `name` et `sourcePath`. Dans l'exemple suivant, l' EC2 instance Amazon hôte contient des données `/ecs/webdata` que vous souhaitez monter dans le conteneur.

```
"volumes": [
  {
    "name": "webdata",
    "host": {
      "sourcePath": "/ecs/webdata"
    }
  }
]
```

2. Dans la section `containerDefinitions`, définissez un conteneur avec une valeur `mountPoints` qui fait référence au nom du montage lié défini et la valeur `containerPath` sur laquelle monter le montage lié sur le conteneur.

```
"containerDefinitions": [
  {
    "name": "web",
    "image": "public.ecr.aws/docker/library/nginx:latest",
    "cpu": 99,
    "memory": 100,
    "portMappings": [
      {
        "containerPort": 80,
        "hostPort": 80
      }
```

```
      ],
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "webdata",
          "containerPath": "/usr/share/nginx/html"
        }
      ]
    }
  ]
```

Pour monter un volume défini sur plusieurs conteneurs à différents emplacements

Vous pouvez définir un volume de données dans une définition de tâche et monter ce volume à différents emplacements de différents conteneurs. Par exemple, votre conteneur hôte possède un dossier de données de site web à l'adresse `/data/webroot`. Vous voudrez peut-être monter ce volume de données en lecture seule sur deux serveurs web qui ont des racines de documents différentes.

1. Dans la section `volumes` de la définition de tâche, définissez un volume de données nommé `webroot` et ayant le chemin source `/data/webroot`.

```
"volumes": [
  {
    "name": "webroot",
    "host": {
      "sourcePath": "/data/webroot"
    }
  }
]
```

2. Dans la section `containerDefinitions`, définissez un conteneur pour chaque serveur web avec des valeurs `mountPoints` qui associent le volume `webroot` à la valeur `containerPath` pointant vers la racine du document pour ce conteneur.

```
"containerDefinitions": [
  {
    "name": "web-server-1",
    "image": "my-repo/ubuntu-apache",
    "cpu": 100,
    "memory": 100,
```

```
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80
        }
      ],
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "webroot",
          "containerPath": "/var/www/html",
          "readOnly": true
        }
      ]
    },
    {
      "name": "web-server-2",
      "image": "my-repo/sles11-apache",
      "cpu": 100,
      "memory": 100,
      "portMappings": [
        {
          "containerPort": 8080,
          "hostPort": 8080
        }
      ],
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "webroot",
          "containerPath": "/srv/www/htdocs",
          "readOnly": true
        }
      ]
    }
  ]
```

Pour monter les volumes à partir d'un autre conteneur à l'aide de **`volumesFrom`**

Pour les tâches hébergées sur des EC2 instances Amazon, vous pouvez définir un ou plusieurs volumes sur un conteneur, puis utiliser le `volumesFrom` paramètre dans une autre définition de conteneur au sein de la même tâche pour monter tous les volumes `sourceContainer` à partir de

leurs points de montage initialement définis. Le paramètre `volumesFrom` s'applique aux volumes définis dans la définition de tâche et à ceux qui sont intégrés à l'image au sein d'un Dockerfile.

1. (Facultatif) Pour partager un volume qui est intégré à une image, suivez les instructions `VOLUME` du fichier Dockerfile. L'exemple de Dockerfile suivant utilise une image `httpd`, puis ajoute un volume et le monte à l'emplacement `dockerfile_volume` dans la racine du document Apache. Il s'agit du dossier utilisé par le serveur web `httpd`.

   ```
   FROM httpd
   VOLUME ["/usr/local/apache2/htdocs/dockerfile_volume"]
   ```

   Vous pouvez créer une image avec ce Dockerfile, la transférer dans un référentiel (comme Docker Hub), puis l'utiliser dans votre définition de tâche. L'`my-repo/httpd_dockerfile_volume`image d'exemple utilisée dans les étapes suivantes a été créée avec le Dockerfile précédent.

2. Créez une définition de tâche qui définit les autres volumes et points de montage pour les conteneurs. Dans la section `volumes` de cet exemple, vous créez un volume vide appelé `empty` qui est géré par le démon Docker. Il existe également un volume hôte défini qui est appelé `host_etc`. Il exporte le dossier `/etc` dans l'instance de conteneur hôte.

   ```
   {
     "family": "test-volumes-from",
     "volumes": [
       {
         "name": "empty",
         "host": {}
       },
       {
         "name": "host_etc",
         "host": {
           "sourcePath": "/etc"
         }
       }
     ],
   ```

   Dans la section des définitions de conteneur, créez un conteneur qui monte les volumes définis précédemment. Dans cet exemple, le conteneur `web` monte les volumes `empty` et `host_etc`. Il s'agit du conteneur qui utilise l'image créée avec un volume dans le Dockerfile.

```
"containerDefinitions": [
    {
      "name": "web",
      "image": "my-repo/httpd_dockerfile_volume",
      "cpu": 100,
      "memory": 500,
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80
        }
      ],
      "mountPoints": [
        {
          "sourceVolume": "empty",
          "containerPath": "/usr/local/apache2/htdocs/empty_volume"
        },
        {
          "sourceVolume": "host_etc",
          "containerPath": "/usr/local/apache2/htdocs/host_etc"
        }
      ],
      "essential": true
    },
```

Créez un autre conteneur qui utilise `volumesFrom` pour monter tous les volumes qui sont associés au conteneur `web`. Tous les volumes du conteneur `web` sont également montés sur le conteneur `busybox`. Cela inclut le volume spécifié dans le Dockerfile qui a été utilisé pour générer l'image `my-repo/httpd_dockerfile_volume`.

```
    {
      "name": "busybox",
      "image": "busybox",
      "volumesFrom": [
        {
          "sourceContainer": "web"
        }
      ],
      "cpu": 100,
      "memory": 500,
      "entryPoint": [
```

```
        "sh",
        "-c"
      ],
      "command": [
        "echo $(date) > /usr/local/apache2/htdocs/empty_volume/date && echo $(date)
 > /usr/local/apache2/htdocs/host_etc/date && echo $(date) > /usr/local/apache2/
htdocs/dockerfile_volume/date"
      ],
      "essential": false
    }
  ]
}
```

Lorsque la tâche est exécutée, les deux conteneurs montent les volumes, et la fonction `command` du conteneur `busybox` écrit la date et l'heure dans un fichier. Ce fichier est appelé `date` dans chacun des dossiers de volumes. Les dossiers sont alors visibles sur le site web affiché par le conteneur `web`.

> **Note**
>
> Étant donné que le conteneur `busybox` exécute une commande rapide avant de s'arrêter, il doit être défini en tant que `"essential": false` dans la définition de conteneur. Dans le cas contraire, il arrête l'ensemble de la tâche lorsqu'il s'arrête.

Gestion de l'espace mémoire d'échange de conteneurs sur Amazon ECS

Avec Amazon ECS, vous pouvez contrôler l'utilisation de l'espace mémoire de swap sur vos instances EC2 Amazon basées sur Linux au niveau du conteneur. En utilisant une configuration d'échange par conteneur, l'échange peut être activé ou désactivé pour chaque conteneur au sein d'une définition de tâche. Pour ceux qui l'ont activé, la quantité maximale d'espace d'échange utilisée peut être limitée. Par exemple, l'échange peut être désactivé dans les conteneurs sensibles à la latence. En revanche, les conteneurs présentant des demandes de mémoire transitoire élevées peuvent avoir le swap activé afin de réduire les risques d' out-of-memoryerreurs lorsque le conteneur est en charge.

La configuration d'échange pour un conteneur est gérée par les paramètres de définition de conteneur suivants :

`maxSwap`

Quantité totale de mémoire d'échange (en Mio) qu'un conteneur peut utiliser. Ce paramètre est traduit en `--memory-swap` option docker run où la valeur est la somme de la mémoire du conteneur plus la `maxSwap` valeur.

Si la valeur `0` est spécifiée pour `maxSwap`, le conteneur n'utilise pas l'échange. Les valeurs acceptées sont `0` ou n'importe quel nombre entier positif. Si le paramètre `maxSwap` n'est pas spécifié, le conteneur utilise la configuration d'échange pour l'instance de conteneur sur laquelle il s'exécute. Une valeur `maxSwap` doit être définie pour que le paramètre `swappiness` soit utilisé.

`swappiness`

Vous pouvez utiliser ce paramètre pour régler le comportement d'échange de mémoire d'un conteneur. Une valeur `swappiness` de `0` fait que l'échange ne se produit pas, sauf si nécessaire. Avec la valeur `swappiness` pour `100`, l'échange de pages a lieu de manière agressive. Les valeurs acceptées sont les nombres entiers compris entre `0` et `100`. Si le paramètre `swappiness` n'est pas spécifié, la valeur par défaut `60` est utilisée. Si aucune valeur n'est spécifiée pour `maxSwap`, le paramètre est ignoré. Ce paramètre correspond à l'`--memory-swappiness`option docker run.

Dans l'exemple suivant, la syntaxe JSON est fournie.

```
"containerDefinitions": [{
        ...
        "linuxParameters": {
            "maxSwap": integer,
            "swappiness": integer
        },
        ...
}]
```

Considérations

Tenez compte des points suivants lorsque vous utilisez une configuration d'échange par conteneur.

- L'espace de swap doit être activé et alloué sur l' EC2 instance Amazon hébergeant vos tâches pour que les conteneurs puissent être utilisés. Par défaut, le swap n'est AMIs pas activé sur les appareils optimisés pour Amazon ECS. Vous devez activer l'échange sur l'instance pour utiliser cette fonction. Pour plus d'informations, consultez Instance Store Swap Volumes dans le guide

de EC2 l'utilisateur Amazon ou Comment allouer de la mémoire pour qu'elle fonctionne comme espace de swap dans une EC2 instance Amazon ? .

- Les paramètres de définition du conteneur d'espace d'échange ne sont pris en charge que pour les définitions de tâches qui spécifient le type de EC2 lancement. Ces paramètres ne sont pas pris en charge pour les définitions de tâches destinées uniquement à l'utilisation d'Amazon ECS sur Fargate.
- Cette fonction est uniquement prise en charge pour les conteneurs Linux. Les conteneurs Windows ne sont pas pris en charge actuellement.
- Si les paramètres de définition de conteneur `maxSwap` et `swappiness` sont omis d'une définition de tâche, chaque conteneur a une valeur `swappiness` par défaut de `60`. De plus, l'utilisation totale du swap est limitée à deux fois la mémoire du conteneur.
- Si vous utilisez des tâches sur Amazon Linux 2023, le paramètre `swappiness` n'est pas pris en charge.

Différences entre les définitions de tâches Amazon ECS pour le type de lancement Fargate

Pour utiliser Fargate, vous devez configurer votre définition de tâche pour utiliser le type de lancement Fargate. Des considérations supplémentaires doivent être prises en compte lors de l'utilisation de Fargate.

Paramètres de définition de tâche

Les tâches qui utilisent le type de lancement Fargate ne prennent pas en charge tous les paramètres de définition de tâche Amazon ECS disponibles. Certains paramètres ne sont pas du tout pris en charge, tandis que d'autres se comportent différemment pour les tâches Fargate.

Les paramètres de définition de tâche suivants ne sont pas valides dans les tâches Fargate :

- `disableNetworking`
- `dnsSearchDomains`
- `dnsServers`
- `dockerSecurityOptions`
- `extraHosts`
- `gpu`

- `ipcMode`
- `links`
- `placementConstraints`
- `privileged`
- `maxSwap`
- `swappiness`

Les paramètres de définition de tâche suivants sont valides dans les tâches Fargate, mais avec des limitations à prendre en considération :

- `linuxParameters` : lorsque vous spécifiez des options propres à Linux appliquées au conteneur, pour `capabilities`, la seule fonctionnalité que vous pouvez ajouter est `CAP_SYS_PTRACE`. Les paramètres `devices`, `sharedMemorySize` et `tmpfs` ne sont pas pris en charge. Pour de plus amples informations, veuillez consulter Paramètres Linux.
- `volumes` : les tâches Fargate prennent en charge uniquement les volumes hôtes de montage lié, le paramètre `dockerVolumeConfiguration` n'est donc pas pris en charge. Pour de plus amples informations, veuillez consulter Volumes.
- `cpu` - Pour les conteneurs Windows sur AWS Fargate, la valeur ne peut pas être inférieure à 1 vCPU.
- `networkConfiguration`- Les tâches Fargate utilisent `awsvpc` toujours le mode réseau.

Afin que votre définition de tâche puisse être utilisée avec Fargate, vous pouvez spécifier ce qui suit lors de l'enregistrement de la définition de tâche :

- Dans le champ AWS Management Console, dans le champ Compatibilités requises, spécifiez`FARGATE`.
- Dans le AWS CLI, spécifiez l'`--requires-compatibilities`option.
- Dans l'API Amazon ECS, spécifiez l'indicateur `requiresCompatibilities`.

Systèmes d'exploitation et architectures

Lorsque vous configurez une tâche et une définition du conteneur pour AWS Fargate, vous devez spécifier le système d'exploitation exécuté par le conteneur. Les systèmes d'exploitation suivants sont pris en charge pour AWS Fargate :

- Amazon Linux 2

> **Note**
>
> Les conteneurs Linux utilisent uniquement le noyau et la configuration du noyau du système d'exploitation hôte. Par exemple, la configuration du noyau inclut les commandes du système `sysctl`. Une image de conteneur Linux peut être créée à partir d'une image de base contenant les fichiers et les programmes de n'importe quelle distribution Linux. Si l'architecture du processeur correspond, vous pouvez exécuter des conteneurs à partir de n'importe quelle image de conteneur Linux sur n'importe quel système d'exploitation.

- Windows Server 2019 Full
- Windows Server 2019 Core
- Windows Server 2022 Full
- Windows Server 2022 Core

Lorsque vous exécutez des conteneurs Windows sur AWS Fargate, vous devez disposer de l'architecture de processeur X86_64.

Lorsque vous exécutez des conteneurs LinuxAWS Fargate, vous pouvez utiliser l'architecture du processeur X86_64 ou l' ARM64 architecture de vos applications basées sur ARM. Pour de plus amples informations, veuillez consulter the section called “Définitions de tâches pour les charges de travail ARM 64 bits”.

UC et mémoire au niveau de la tâche

Pour les définitions de tâche Amazon ECS pour AWS Fargate, le CPU et la mémoire doivent être spécifiés au niveau de la tâche. La plupart des cas d'utilisation requièrent uniquement la spécification de ces ressources au niveau de la tâche. Le tableau suivant indique les combinaisons valides d'UC et de mémoire au niveau de la tâche. Vous pouvez spécifier des valeurs de mémoire dans la définition de tâche sous forme de chaîne en MiB ou en Go. Par exemple, vous pouvez spécifier une valeur de mémoire `3072` en MiB ou `3 GB` en Go. Vous pouvez spécifier les valeurs du processeur dans le fichier JSON sous forme de chaîne en unités de processeur ou en mode virtuel CPUs (vCPUs). Par exemple, vous pouvez spécifier une valeur de processeur soit `1024` en unités de processeur, soit `1 vCPU` en CPUs v.

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
256 (0,25 vCPU)	512 Mio, 1 Go, 2 Go	Linux
512 (0,5 vCPU)	1 Go, 2 Go, 3 Go, 4 Go	Linux
1 024 (1 vCPU)	2 Go, 3 Go, 4 Go, 5 Go, 6 Go, 7 Go, 8 Go	Linux, Windows
2 048 (2 vCPU)	Entre 4 Go et 16 Go par incréments de 1 Go	Linux, Windows
4 096 (4 vCPU)	Entre 8 Go et 30 Go par incréments de 1 Go	Linux, Windows
8192 (8 vCPU) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 16 Go et 60 Go par incréments de 4 Go	Linux
16384 (16vCPU) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 32 Go et 120 Go par incréments de 8 Go	Linux

Mise en réseau des tâches

Les tâches Amazon ECS pour AWS Fargate nécessitent le mode réseau `awsvpc`, qui fournit à chaque tâche une interface réseau Elastic. Lorsque vous exécutez une tâche ou créez un service

avec ce mode réseau, vous devez spécifier un ou plusieurs sous-réseaux pour attacher l'interface réseau et un ou plusieurs groupes de sécurité à appliquer à l'interface réseau.

Si vous utilisez des sous-réseaux publics, déterminez s'il est nécessaire de fournir une adresse IP publique pour l'interface réseau. Pour qu'une tâche Fargate dans un sous-réseau public puisse extraire des images des conteneurs, une adresse IP publique doit être attribuée à l'interface réseau Elastic de la tâche, avec une route vers Internet ou une passerelle NAT pouvant acheminer les demandes vers Internet. Le sous-réseau nécessite qu'une passerelle NAT soit attachée aux demandes d'acheminement à Internet pour qu'une tâche Fargate dans un sous-réseau privé puisse extraire des images de conteneur. Lorsque vous hébergez vos images de conteneur dans Amazon ECR, vous pouvez configurer Amazon ECR pour utiliser un point de terminaison VPC d'interface. Dans ce cas, l' IPv4 adresse privée de la tâche est utilisée pour l'extraction de l'image. Pour plus d'informations sur les points de terminaison d'interface Amazon ECR, consultez Points de terminaison d'un VPC d'interface Amazon ECR (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.

Voici un exemple de `networkConfiguration` section pour un service Fargate :

```
"networkConfiguration": {
   "awsvpcConfiguration": {
      "assignPublicIp": "ENABLED",
      "securityGroups": [ "sg-12345678" ],
      "subnets": [ "subnet-12345678" ]
   }
}
```

Limites des ressources de tâche

Les définitions de tâche Amazon ECS pour Linux sur AWS Fargate prennent en charge le paramètre `ulimits` permettant de définir les limites de ressources d'un conteneur.

Les définitions de tâche Amazon ECS pour Windows sur AWS Fargate ne prennent pas en charge le paramètre `ulimits` permettant de définir les limites de ressources d'un conteneur.

Les tâches Amazon ECS hébergées sur Fargate utilisent les valeurs de limite définies par défaut par le système d'exploitation, à l'exception du paramètre de limite de ressources `nofile`. La limite de ressources `nofile` restreint le nombre de fichiers ouverts qu'un conteneur peut utiliser. Sur Fargate, la limite flexible par défaut `nofile` est `65535` et la limite stricte est `65535`. Vous pouvez définir les valeurs des deux limites jusqu'à `1048576`.

L'exemple suivant présente un extrait de définition de tâche qui montre comment définir une limite `nofile` personnalisée qui a été doublée :

```
"ulimits": [
    {
       "name": "nofile",
       "softLimit": 2048,
       "hardLimit": 8192
    }
]
```

Pour plus d'informations sur les autres limites de ressources pouvant être ajustées, consultez Limites des ressources.

Journalisation

Journalisation des événements

Amazon ECS enregistre les actions qu'il entreprend EventBridge. Vous pouvez utiliser les événements Amazon ECS EventBridge pour recevoir des notifications en temps quasi réel concernant l'état actuel de vos clusters, services et tâches Amazon ECS. Vous pouvez également automatiser des actions pour répondre à ces événements. Pour de plus amples informations, veuillez consulter Automatisez les réponses aux erreurs Amazon ECS à l'aide de EventBridge.

Journalisation du cycle de vie des tâches

Les tâches exécutées sur Fargate publient des horodatages pour suivre l'état de la tâche au cours de son cycle de vie. Vous pouvez voir les horodatages dans les détails de la tâche dans le AWS Management Console et en décrivant la tâche dans le AWS CLI et. SDKs Par exemple, vous pouvez utiliser les horodatages pour évaluer le temps passé par la tâche à télécharger les images du conteneur et décider si vous devez optimiser la taille de l'image du conteneur ou utiliser des index Seekable OCI. Pour plus d'informations sur les pratiques relatives aux images de conteneur, veuillez consulter Bonnes pratiques pour les images de conteneurs Amazon ECS.

Journalisation d'applications

Les définitions de tâche Amazon ECS pour AWS Fargate prennent en charge uniquement les pilotes de journal `awslogs`, `splunk` et `awsfirelens` pour la configuration de journal.

Le pilote de `awslogs` journal configure vos tâches Fargate pour envoyer des informations de journal à Amazon Logs. CloudWatch L'exemple suivant montre un extrait d'une définition de tâche dans lequel le pilote de journal `awslogs` est configuré :

```
"logConfiguration": {
   "logDriver": "awslogs",
   "options": {
      "awslogs-group" : "/ecs/fargate-task-definition",
      "awslogs-region": "us-east-1",
      "awslogs-stream-prefix": "ecs"
   }
}
```

Pour plus d'informations sur l'utilisation du pilote de `awslogs` journal dans une définition de tâche pour envoyer les journaux de vos conteneurs à CloudWatch Logs, consultezEnvoyez les journaux Amazon ECS à CloudWatch .

Pour plus d'informations sur l'utilisation du pilote de journal `awsfirelens` dans une définition de tâche, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

Pour plus d'informations sur l'utilisation du pilote de journal `splunk` dans une définition de tâche, consultez `splunk`pilote de journal.

Stockage de tâche

Pour les tâches Amazon ECS hébergées sur Fargate, les types de stockage suivants sont pris en charge :

- Les volumes Amazon EBS fournissent un stockage par blocs rentable, durable et performant pour les charges de travail conteneurisées gourmandes en données. Pour de plus amples informations, veuillez consulter Utiliser des volumes Amazon EBS avec Amazon ECS.
- Volumes Amazon EFS pour le stockage permanent. Pour de plus amples informations, veuillez consulter Utiliser les volumes Amazon EFS avec Amazon ECS.
- Supports de liaison pour un stockage éphémère. Pour de plus amples informations, veuillez consulter Utiliser des montages par liaison avec Amazon ECS.

Chargement différé d'images de conteneurs à l'aide de Seekable OCI (SOCI)

Les tâches Amazon ECS sur Fargate qui utilisent la version de plateforme `1.4.0` de Linux peuvent utiliser Seekable OCI (SOCI) pour accélérer le démarrage des tâches. Avec SOCI, les conteneurs ne passent que quelques secondes à extraire l'image avant de pouvoir démarrer, ce qui laisse le temps de configurer l'environnement et d'instancier l'application pendant que l'image est téléchargée en arrière-plan. C'est ce qu'on appelle le chargement différé. Lorsque Fargate lance une tâche Amazon ECS, il détecte automatiquement si un index SOCI existe pour une image de la tâche et démarre le conteneur sans attendre que l'image complète soit téléchargée.

Pour les conteneurs qui s'exécutent sans index SOCI, les images des conteneurs sont entièrement téléchargées avant le démarrage du conteneur. Ce comportement est le même sur toutes les autres versions de plate-forme de Fargate et sur l'AMI optimisée pour Amazon ECS sur les instances Amazon. EC2

Seekable OCI (SOCI) est une technologie open source développée par AWS qui permet de lancer des conteneurs plus rapidement en chargeant paresseusement l'image du conteneur. SOCI fonctionne en créant un index (index SOCI) des fichiers dans une image de conteneur existante. Cet index permet de lancer les conteneurs plus rapidement, en offrant la possibilité d'extraire un fichier individuel d'une image de conteneur avant de télécharger l'image complète. L'index SOCI doit être stocké sous forme d'artefact dans le même référentiel que l'image dans le registre des conteneurs. Vous ne devez utiliser que les index SOCI provenant de sources fiables, car l'index est la source officielle du contenu de l'image. Pour plus d'informations, veuillez consulter Présentation de Seekable OCI pour les images de conteneur à chargement différé.

Les clients souhaitant utiliser SOCI ne pourront utiliser que le manifeste d'index SOCI v2. Les clients existants qui ont déjà utilisé SOCI sur Fargate peuvent continuer à utiliser le SOCI Index Manifest v1, mais nous leur conseillons vivement de migrer vers le SOCI Index Manifest v2. Le manifeste d'index SOCI v2 crée une relation explicite entre les images de conteneurs et leurs index SOCI afin de garantir des déploiements cohérents.

Considérations

Si vous souhaitez que Fargate utilise un index SOCI pour charger des images de conteneur de manière différée dans une tâche, prenez en compte les éléments suivants :

- Seules les tâches exécutées sur une version de plateforme Linux `1.4.0` peuvent utiliser les index SOCI. Les tâches exécutées sur des conteneurs Windows sur Fargate ne sont pas prises en charge.

- Les tâches exécutées sur une architecture de processeur X86_64 ou ARM64 sont prises en charge.
- Les images de conteneur incluses dans la définition de tâche doivent être stockées dans un registre d'images compatible. Voici une liste des registres compatibles :
  - Registres privés Amazon ECR.
- Seules les images de conteneur qui utilisent gzip la compression ou qui ne sont pas compressées sont prises en charge. Les images de conteneur qui utilisent la compression zstd ne sont pas prises en charge.
- Pour le manifeste d'index SOCI v2, la génération d'un manifeste d'index SOCI modifie le manifeste de l'image du conteneur lorsque nous ajoutons une annotation pour l'index SOCI. Cela se traduit par un nouveau condensé d'image du conteneur. Le contenu des couches du système de fichiers contenant des images ne change pas.
- Pour SOCI Index Manifest v2, lorsque l'image du conteneur a déjà été stockée dans le référentiel d'images du conteneur, une fois qu'un index SOCI a été généré, vous devez republier l'image du conteneur. La rediffusion d'une image de conteneur n'augmente pas les coûts de stockage en dupliquant les couches du système de fichiers, elle télécharge uniquement un nouveau fichier manifeste.
- Nous vous recommandons d'essayer le chargement différé avec des images de conteneur dont la taille est supérieure à 250 MiB compressés. Il est moins probable que le temps nécessaire pour charger des images plus petites soit réduit.
- Le chargement différé pouvant modifier le temps de démarrage de vos tâches, vous devrez peut-être modifier différents délais, tels que le délai de grâce de surveillance de l'état pour Elastic Load Balancing.
- Si vous souhaitez empêcher le chargement différé d'une image de conteneur, l'image de conteneur devra être republiée sans index SOCI attaché.

Création d'un index Seekable OCI

Pour qu'une image de conteneur soit chargée en différé, un index SOCI (un fichier de métadonnées) doit être créé et stocké dans le référentiel d'images de conteneur à côté de l'image de conteneur. Pour créer et envoyer un index SOCI, vous pouvez utiliser l'outil open source soci-snapshotter CLI sur. GitHub Vous pouvez également déployer le générateur d'index CloudFormation AWS SOCI. Il s'agit d'une solution sans serveur qui crée et transmet automatiquement un index SOCI lorsqu'une image de conteneur est transmise à Amazon ECR. Pour plus d'informations sur la solution et les étapes d'installation, consultez CloudFormation AWS SOCI Index Builder sur GitHub. Le

CloudFormation AWS SOCI Index Builder est un moyen d'automatiser le démarrage avec SOCI, tandis que l'outil SOCI open source offre une plus grande flexibilité en matière de génération d'index et permet d'intégrer la génération d'index dans vos pipelines d'intégration continue et de livraison continue (CI/CD).

> **Note**
>
> Pour que l'index SOCI soit créé pour une image, celle-ci doit exister dans le magasin d'images containerd de l'ordinateur exécutant `soci-snapshotter`. Si l'image se trouve dans le magasin d'images Docker, elle est introuvable.

Vérification de l'utilisation du chargement différé par une tâche

Pour vérifier qu'une tâche a été chargée en différé à l'aide de SOCI, vérifiez le point de terminaison des métadonnées de la tâche depuis cette dernière. Lorsque vous interrogez le point de terminaison des métadonnées de tâche version 4, il existe un champ `Snapshotter` dans le chemin par défaut du conteneur à partir duquel vous interrogez. De plus, il existe des champs `Snapshotter` pour chaque conteneur du chemin `/task`. La valeur par défaut de ce champ est `overlayfs`, et ce champ est défini sur `soci` si SOCI est utilisé. Pour vérifier qu'un manifeste d'index SOCI v2 est joint à une image de conteneur, vous pouvez récupérer l'index d'image auprès d'Amazon ECR à l'aide du. AWS CLI

```
IMAGE_REPOSITORY=r
IMAGE_TAG=latest

aws ecr batch-get-image \
    --repository-name=$IMAGE_REPOSITORY \
    --image-ids imageTag=$IMAGE_TAG \
    --query 'images[0].imageManifest' --output text | jq -r '.manifests[] |
 select(.artifactType=="application/vnd.amazon.soci.index.v2+json")'
```

Pour vérifier qu'un manifeste d'index SOCI v1 est attaché à une image de conteneur, vous pouvez utiliser l'API OCI Referrers.

```
ACCOUNT_ID=111222333444
AWS_REGION=us-east-1
IMAGE_REPOSITORY=nginx-demo
```

```
IMAGE_TAG=latest
IMAGE_DIGEST=$(aws ecr describe-images --repository-name $IMAGE_REPOSITORY --image-ids
 imageTag=$IMAGE_TAG --query 'imageDetails[0].imageDigest' --output text)
ECR_PASSWORD=$(aws ecr get-login-password)

curl \
    --silent \
    --user AWS:$ECR_PASSWORD \
    https://$ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/v2/$IMAGE_REPOSITORY/
referrers/$IMAGE_DIGEST?artifactType=application%2Fvnd.amazon.soci.index.v1%2Bjson | jq
 -r '.'
```

Différences de définition des tâches Amazon ECS pour EC2 les instances exécutant Windows

Les tâches exécutées sur des instances EC2 Windows ne prennent pas en charge tous les paramètres de définition de tâches Amazon ECS disponibles. Certains paramètres ne sont pas du tout pris en charge, tandis que d'autres se comportent différemment.

Les paramètres de définition de tâches suivants ne sont pas pris en charge pour les définitions de tâches Amazon EC2 Windows :

- `containerDefinitions`
  - `disableNetworking`
  - `dnsServers`
  - `dnsSearchDomains`
  - `extraHosts`
  - `links`
  - `linuxParameters`
  - `privileged`
  - `readonlyRootFilesystem`
  - `user`
  - `ulimits`
- `volumes`
  - `dockerVolumeConfiguration`
- `cpu`

Nous vous recommandons de spécifier une UC de niveau conteneur pour les conteneurs Windows.

- `memory`

  Nous vous recommandons de spécifier une mémoire de niveau conteneur pour les conteneurs Windows.

- `proxyConfiguration`
- `ipcMode`
- `pidMode`
- `taskRoleArn`

  Les fonctionnalités des rôles IAM pour les tâches sur les instances EC2 Windows nécessitent une configuration supplémentaire, mais une grande partie de cette configuration est similaire à la configuration des rôles IAM pour les tâches sur les instances de conteneur Linux. Pour de plus amples informations, veuillez consulter the section called “ Configuration supplémentaire de l'instance Amazon EC2 Windows”.

Création d'une définition de tâche Amazon ECS à l'aide de la console

Vous créez une définition de tâche afin de définir l'application que vous exécutez en tant que tâche ou service.

Lorsque vous créez une définition de tâche pour le type de lancement externe, vous devez créer la définition de tâche à l'aide de l'éditeur JSON et définir le `requireCapabilities` paramètre sur`EXTERNAL`.

Vous pouvez créer une définition de tâche en utilisant l'expérience de la console ou en spécifiant un fichier JSON.

Validation JSON

L'éditeur JSON de la console Amazon ECS valide les éléments suivants dans le fichier JSON :

- Le fichier est un fichier JSON valide.
- Le fichier ne contient aucune clé superflue.

- Le fichier contient le `familyName` paramètre.
- Il y a au moins une entrée en dessous `decontainerDefinitions`.

AWS CloudFormation piles

Le comportement suivant s'applique aux définitions de tâches créées dans la nouvelle console Amazon ECS avant le 12 janvier 2023.

Lorsque vous créez une définition de tâche, la console Amazon ECS crée automatiquement une CloudFormation pile dont le nom commence par`ECS-Console-V2-TaskDefinition-`. Si vous avez utilisé le AWS CLI ou un AWS SDK pour annuler l'enregistrement de la définition de tâche, vous devez supprimer manuellement la pile de définitions de tâches. Pour plus d'informations, consultez la section Suppression d'une pile dans le guide de AWS CloudFormation l'utilisateur.

Aucune CloudFormation pile n'est créée automatiquement pour les définitions de tâches créées après le 12 janvier 2023.

Procédure

Amazon ECS console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Dans le menu Créer une nouvelle définition de tâche, choisissez Créer une nouvelle définition de tâche.
4. Pour Task definition family (Famille de définition de tâche), spécifiez un nom unique pour la définition de tâche.
5. Pour Type de lancement, choisissez l'environnement de l'application. La console par défaut est AWS Fargate(qui est sans serveur). Amazon ECS utilise cette valeur pour effectuer une validation afin de garantir que les paramètres de définition des tâches sont valides pour le type d'infrastructure.
6. Pour Operating system/Architecture (Système d'exploitation/architecture), choisissez le système d'exploitation et l'architecture du processeur pour la tâche.

   Pour exécuter votre tâche sur une architecture ARM 64 bits, choisissez Linux/ ARM64. Pour de plus amples informations, veuillez consulter the section called “Plateforme d'exécution”.

Pour exécuter vos tâches AWS Fargate sur les conteneurs Windows, choisissez un système d'exploitation Windows pris en charge. Pour de plus amples informations, veuillez consulter the section called “Systèmes d'exploitation et architectures”.

7. Pour Task size (Taille de la tâche), choisissez les valeurs du processeur et de la mémoire à réserver pour la tâche. La valeur du processeur est spécifiée comme v CPUs et la mémoire est spécifiée comme Go.

   Pour les tâches hébergées sur Fargate, le tableau suivant indique les combinaisons de processeur et de mémoire valides.

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
256 (0,25 vCPU)	512 Mio, 1 Go, 2 Go	Linux
512 (0,5 vCPU)	1 Go, 2 Go, 3 Go, 4 Go	Linux
1 024 (1 vCPU)	2 Go, 3 Go, 4 Go, 5 Go, 6 Go, 7 Go, 8 Go	Linux, Windows
2 048 (2 vCPU)	Entre 4 Go et 16 Go par incréments de 1 Go	Linux, Windows
4 096 (4 vCPU)	Entre 8 Go et 30 Go par incréments de 1 Go	Linux, Windows
8192 (8 vCPU) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 16 Go et 60 Go par incréments de 4 Go	Linux

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
16384 (16vCPU) Note Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 32 Go et 120 Go par incréments de 8 Go	Linux

Pour les tâches utilisant le type de lancement EC2 ou externe, les valeurs de processeur de tâche prises en charge sont comprises entre 128 unités de processeur (0,125 vCPUs) et 196 608 unités de processeur (192 vCPUs).

Pour spécifier la valeur de la mémoire en Go, entrez Go après la valeur. Par exemple, pour définir la valeur de la mémoire sur 3 Go, entrez 3 Go.

> **Note**
>
> Les paramètres d'UC et de mémoire de niveau tâche sont ignorés pour les conteneurs Windows.

8. Pour Network mode (Mode réseau), choisissez le mode réseau à utiliser. La valeur par défaut est le mode awsvpc. Pour plus d'informations, consultez Réseaux des tâches Amazon ECS.

   Si vous choisissez bridge, sous Mappages de ports, pour Port hôte, entrez le numéro de port sur l'instance de conteneur à réserver pour votre conteneur.

9. (Facultatif) Développez la section Rôles des tâches pour configurer les rôles AWS Identity and Access Management (IAM) de la tâche :

   a. Pour Task role (Rôle de tâche), choisissez le rôle IAM à assigner à la tâche. Un rôle IAM de tâche autorise les conteneurs d'une tâche à appeler des opérations d' AWS API.

   b. Pour Rôle d'exécution de tâche, choisissez le rôle.

Pour savoir quand utiliser un rôle d'exécution de tâche, veuillez consulter the section called “Rôle IAM d'exécution de tâche”. Si vous n'avez pas besoin du rôle, choisissez Aucun.

10. (Facultatif) Développez la section Placement des tâches pour ajouter des contraintes de placement. Les contraintes de placement des tâches vous permettent de filtrer les instances de conteneur utilisées pour le placement de vos tâches à l'aide d'attributs intégrés ou personnalisés.
11. (Facultatif) Développez la section d'injection de défauts pour activer l'injection de défauts. L'injection de défauts vous permet de tester la façon dont votre application répond à certains scénarios de défaillance.
12. Pour chaque conteneur à définir dans votre définition de tâche, effectuez les étapes suivantes.

    a. Pour Name (Nom), saisissez un nom pour le conteneur.

    b. Pour Image URI (URI de l'image), saisissez l'image à utiliser pour démarrer un conteneur. Les images du registre Amazon ECR Public Gallery peuvent être spécifiées en utilisant uniquement le nom du registre Amazon ECR Public. Par exemple, si cela `public.ecr.aws/ecs/amazon-ecs-agent:latest` est spécifié, le conteneur Amazon Linux hébergé sur la galerie publique Amazon ECR est utilisé. Pour tous les autres référentiels, spécifiez le référentiel en utilisant le `repository-url/image@digest` format `repository-url/image:tag` or.

    c. Si votre image se trouve dans un registre privé en dehors d'Amazon ECR, sous Registre privé, activez Authentification du registre privé. Ensuite, dans ARN ou nom Secrets Manager, saisissez l'Amazon Resource Name (ARN) du secret.

    d. Pour le conteneur essentiel, si votre définition de tâche comporte deux conteneurs ou plus, vous pouvez spécifier si le conteneur doit être considéré comme essentiel. Lorsqu'un conteneur est marqué comme essentiel, si ce conteneur s'arrête, la tâche est arrêtée. Chaque définition de tâche doit contenir au moins un conteneur essentiel.

    e. Un mappage de ports permet aux conteneurs d'accéder aux ports de l'hôte pour envoyer ou recevoir du trafic. Sous Port mappings (Mappages de port), effectuez l'une des actions suivantes :

        - Lorsque vous utilisez le mode réseau awsvpc, pourContainer port (Port du conteneur) et Protocol (Protocole), choisissez le mappage de port à utiliser pour le conteneur.

- Lorsque vous utilisez le mode réseau bridge (pont), pour Container port (Port du conteneur) et Protocol (Protocole), choisissez le mappage de port à utiliser pour le conteneur.

Choisissez Add more port mappings (Ajouter d'autres mappages de ports) pour spécifier des mappages de ports de conteneur supplémentaires.

f. Pour donner au conteneur un accès en lecture seule à son système de fichiers racine, pour Système de fichiers racine en lecture seule, sélectionnez Lecture seule.

g. (Facultatif) Pour définir les limites de CPU, de GPU et de mémoire au niveau du conteneur qui sont différentes des valeurs au niveau des tâches, sous Limites d'allocation de ressources, procédez comme suit :

- Pour le processeur, entrez le nombre d'unités de processeur que l'agent de conteneur Amazon ECS réserve pour le conteneur.
- Dans GPU, entrez le nombre d'unités GPU pour l'instance de conteneur.

  Une EC2 instance Amazon compatible avec le GPU dispose d'une unité GPU pour chaque GPU. Pour de plus amples informations, veuillez consulter the section called “Définitions de tâches pour les charges de travail du GPU”.
- Pour Limite matérielle de mémoire, entrez la quantité de mémoire, en Go, à présenter au conteneur. Si le conteneur tente de dépasser la limite stricte, il s'arrête.
- Le démon Docker 20.10.0 ou version ultérieure réserve un minimum de 6 mégaoctets (Mo) de mémoire pour un conteneur. Ne spécifiez donc pas moins de 6 Mo de mémoire pour vos conteneurs.

  Le démon Docker 19.03.13-ce ou une version antérieure réserve un minimum de 4 Mo de mémoire pour un conteneur. Ne spécifiez donc pas moins de 4 Mo de mémoire pour vos conteneurs.
- Pour Limite flexible de mémoire, saisissez la limite flexible (en Go) de mémoire à réserver pour le conteneur.

  En cas de contention de la mémoire système, Docker tente de garder la mémoire du conteneur en deçà de cette limite flexible. Si vous ne spécifiez pas de mémoire au niveau de la tâche, vous devez indiquer un nombre entier différent de zéro pour Limite stricte de mémoire et/ou Limite flexible de mémoire. Si vous spécifiez les deux, Limite stricte de mémoire doit être supérieure à Limite flexible de mémoire.

Cette fonctionnalité n'est pas prise en charge sur les conteneurs Windows.

h. (Facultatif) Développez la section Variables d'environnement pour spécifier les variables d'environnement à injecter dans le conteneur. Vous pouvez spécifier des variables d'environnement soit individuellement en utilisant des paires clé-valeur, soit en bloc en spécifiant un fichier de variables d'environnement hébergé dans un compartiment Amazon S3. Pour plus d'informations sur le formatage d'un fichier de variables d'environnement, consultezTransmettre une variable d'environnement individuelle à un conteneur Amazon ECS.

   Lorsque vous spécifiez une variable d'environnement pour le stockage secret, entrez le nom du secret dans Key. Ensuite ValueFrom, entrez l'ARN complet du secret Parameter Store ou du secret Secrets Manager

i. (Facultatif) Sélectionnez l'option Use log collection (Utiliser la collecte de journaux) pour spécifier une configuration de journal. Pour chaque pilote de journal disponible, il existe des options de pilote de journal à spécifier. L'option par défaut envoie les journaux des conteneurs à Amazon CloudWatch Logs. Les autres options du pilote de journal sont configurées à l'aide de AWS FireLens. Pour de plus amples informations, veuillez consulter Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

   Voici une description plus détaillée de chaque destination de journal de conteneur.

   - Amazon CloudWatch — Configurez la tâche pour envoyer les journaux des conteneurs à CloudWatch Logs. Les options du pilote de journal par défaut sont fournies, ce qui permet de créer un groupe de CloudWatch journaux en votre nom. Pour spécifier un autre nom de groupe de journaux, modifiez les valeurs des options de pilote.
   - Exporter les journaux vers Splunk : configurez la tâche pour envoyer les journaux des conteneurs au Splunk pilote qui envoie les journaux à un service distant. Vous devez saisir l'URL de votre service Splunk Web. Le Splunk jeton est spécifié comme option secrète car il peut être traité comme une donnée sensible.
   - Exporter les journaux vers Amazon Data Firehose : configurez la tâche pour envoyer les journaux des conteneurs à Firehose. Les options du pilote de journal par défaut sont fournies, qui envoie le journal à un flux de diffusion Firehose. Pour spécifier un autre nom de flux de diffusion, modifiez les valeurs des options de pilote.
   - Exporter les journaux vers Amazon Kinesis Data Streams : configurez la tâche pour envoyer les journaux des conteneurs à Kinesis Data Streams. Les options du pilote de journal par défaut sont fournies, qui envoient les journaux vers un flux Kinesis Data

Streams. Pour spécifier un autre nom de flux, modifiez les valeurs des options de pilote.

- Exporter les journaux vers Amazon OpenSearch Service : configurez la tâche pour envoyer les journaux des conteneurs vers un domaine OpenSearch de service. Les options de pilote de journal doivent être fournies.
- Exporter les journaux vers Amazon S3 : configurez la tâche pour envoyer les journaux des conteneurs vers un compartiment Amazon S3. Les options du pilote de journal par défaut sont fournies, mais vous devez spécifier un nom de compartiment Amazon S3 valide.

j. (Facultatif) Configurez des paramètres de conteneur supplémentaires.

Pour configurer cette option	Faites ceci	
Politique de redémarrage Ces options définissent une politique de redémarrage pour redémarrer un conteneur à sa sortie.	Développez la politique de redémarrage, puis configurez les éléments suivants : • Pour activer une politique de redémarrage pour le conteneur, activez Activer la politique de redémarrage. • Pour les codes de sortie ignorés, spécifiez une liste de codes de sortie de conteneur entiers séparés par des virgules. Si le conteneur sort avec l'un des codes de sortie spécifiés, Amazon ECS n'essaiera pas de redémarrer le conteneur. Si rien n'est spécifié, Amazon ECS n'ignorera aucun code de sortie. • Pour Période de tentative de réinitialisation, spécifiez une période entière, en secondes, pendant	

Pour configurer cette option	Faites ceci	
	laquelle le conteneur doit fonctionner avant de pouvoir tenter de redémarrer en cas de sortie. Amazon ECS ne peut tenter de redémarrer un conteneur qu'une seule fois par période de tentative de réinitialisation (secondes). Si rien n'est spécifié, le conteneur doit fonctionner pendant 300 secondes avant de pouvoir tenter de redémarrer.	

Pour configurer cette option	Faites ceci	
HealthCheck Ce sont les commandes qui déterminent si un conteneur est sain. Pour de plus amples informations, veuillez consulter Déterminer l'état des tâches Amazon ECS à l'aide de vérifications de l'état des conteneurs.	Développez HealthCheck, puis configurez les éléments suivants : • Pour Command (Commande), saisissez une liste de commandes séparées par des virgules. Vous pouvez démarrer les commandes par `CMD` pour exécuter directement les arguments de la commande, ou par `CMD-SHELL` pour exécuter la commande avec le shell par défaut du conteneur. Si vous n'en spécifiez aucun, `CMD` est utilisé. • Pour Interval (Intervalle), entrez le nombre de secondes entre chaque surveillance de l'état. Les valeurs valides sont comprises entre 5 et 30. • Pour Timeout (Délai), entrez la durée (en secondes) d'attente pour qu'une surveillance de l'état réussisse avant	

Pour configurer cette option	Faites ceci	
	qu'elle ne soit considérée comme un échec. Les valeurs valides sont comprises entre 2 et 60. • Pour Start period (Période de démarrage), entrez la durée (en secondes) d'attente pour qu'un conteneur s'amorce avant que les commandes de surveillance de l'état s'exécutent. Les valeurs valides sont comprises entre 0 et 300. • Pour Retries (Tentatives), entrez le nombre de tentatives d'exécution des commandes de surveillance de l'état en cas d'échec. Les valeurs valides sont comprises entre 1 et 10.	

Pour configurer cette option	Faites ceci	
Ordre des dépendances au démarrage Cette option définit les dépendances pour le démarrage et l'arrêt des conteneurs. Un conteneur peut contenir plusieurs dépendances.	Développez Ordre des dépendances de démarrage, puis configurez les éléments suivants : a. Choisissez Ajouter une dépendance de conteneur. b. Pour Conteneur, choisissez le conteneur. c. Pour Condition, choisissez la condition de dépendance de démarrage. Pour ajouter une dépendance supplémentaire, choisissez Ajouter une dépendance de conteneur.	

Pour configurer cette option	Faites ceci	
Délais de conteneur Ces options déterminent à quel moment démarrer et arrêter un conteneur.	Développez Délais de conteneur, puis configurez les éléments suivants : • Pour configurer le temps d'attente avant de renoncer à résoudre les dépendances d'un conteneur, entrez le nombre de secondes dans le champ Start timeout. • Pour configurer le temps d'attente avant que le conteneur ne soit arrêté s'il ne sort pas normalement tout seul, pour Stop timeout, entrez le nombre de secondes.	

Pour configurer cette option	Faites ceci	
Paramètres réseau de conteneur Ces options détermine nt s'il faut utiliser la mise en réseau au sein d'un conteneur.	Développez Paramètres réseau de conteneur, puis configurez les éléments suivants : • Pour désactiver la mise en réseau des conteneurs, sélection nez Désactiver le réseau. • Pour configure r les adresses IP des serveurs DNS présentées au conteneur, dans Serveurs DNS, saisissez l'adresse IP de chaque serveur sur une ligne distincte. • Pour configurer les domaines DNS afin de rechercher les noms d'non-fully-qualified hôtes présentés au conteneur, dans les domaines de recherche DNS, entrez chaque domaine sur une ligne distincte.	

Pour configurer cette option	Faites ceci	
	Le schéma est **`^[a-zA-Z0-9-.]{0,253}[a-zA-Z0-9]$`** . • Pour configurer le nom d'hôte du conteneur , dans Nom d'hôte, saisissez le nom d'hôte du conteneur. • Pour ajouter des noms d'hôte et des mappages d'adresses IP ajoutés au fichier `/etc/hosts` sur le conteneur, choisissez Ajouter un hôte supplémentaire, puis pour Nom d'hôte et Adresse IP, saisissez le nom d'hôte et l'adresse IP.	

Pour configurer cette option	Faites ceci	
Configuration Docker Elles remplacent les valeurs duDockerfile.	Développez Dockerla configuration, puis configurez les éléments suivants : • Dans le champ Commande, entrez une commande exécutable pour un conteneur. Ce paramètre correspond `Cmd` à la section Créer un conteneur de l'API Docker Remote et à l'`COMMAND`option de`docker run`. Ce paramètre remplace l'`CMD`instruction contenue dans un Dockerfile. • Pour Point d'entrée, entrez le Docker POINT D'ENTRÉE transmis au conteneur. Ce paramètre correspond `Entrypoint` à la section Créer un conteneur de l'API Docker Remote et à l'`--entrypoint` option de`docker run`.	

Pour configurer cette option	Faites ceci	
	Ce paramètre remplace l'`ENTRYPOINT` instruction contenue dans un Dockerfile. • Dans le champ Répertoire de travail, entrez le répertoire dans lequel le conteneur exécutera tout point d'entrée et les instructions de commande fournies. Ce paramètre correspond `WorkingDir` à la section Créer un conteneur de l'API Docker Remote et à l'`--workdir` option de `docker run`. Ce paramètre remplace l'`WORKDIR` instruction contenue dans un Dockerfile.	

Pour configurer cette option	Faites ceci
Limites de ressources (Ulimits) Ces valeurs remplacent le paramètre de quota de ressources par défaut pour le système d'exploitation. Ce paramètre correspond à `Ulimits` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--ulimit` correspond à docker run.	Développez les limites de ressources (ulimits), puis choisissez Ajouter ulimit. Dans Nom de la limite, choisissez la limite. Ensuite, pour Limite flexible et Limite stricte, saisissez les valeurs. Pour en ajouter d'autres ulimits, choisissez Ajouter ulimit.
Dockerétiquettes Cette option ajoute des métadonnées à votre conteneur. Ce paramètre correspond à `Labels` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--label` correspond à docker run.	Développez Dockerles étiquettes, choisissez Ajouter une paire clé-valeur, puis entrez la clé et la valeur. Pour ajouter des Docker libellés supplémentaires, choisissez Ajouter une paire clé-valeur.

k. (Facultatif) Choisissez Add more containers (Ajouter des conteneurs supplémentaires) pour ajouter des conteneurs supplémentaires à la définition de tâche.

13. (Facultatif) La section Stockage est utilisée pour augmenter la quantité de stockage éphémère pour les tâches hébergées sur Fargate. Vous pouvez également utiliser cette section pour ajouter une configuration de volume de données pour la tâche.

    - Pour étendre le stockage éphémère disponible au-delà de la valeur par défaut de 20 gibibytes (GiB) pour vos tâches Fargate pour Amount (Quantité), spécifiez une valeur allant jusqu'à 200 GiB.

14. (Facultatif) Pour ajouter une configuration de volume de données pour la définition de tâche, choisissez Ajouter un volume, puis procédez comme suit.

    a. Pour Volume name (Nom du volume), saisissez un nom pour le volume de données. Le nom du volume de données est utilisé lors de la création d'un point de montage de conteneur.

    b. Pour la configuration du volume, indiquez si vous souhaitez configurer votre volume lors de la création de la définition de tâche ou lors du déploiement.

    > **Note**
    >
    > Les volumes qui peuvent être configurés lors de la création d'une définition de tâche incluent Bind mountDocker, Amazon EFS et Amazon FSx pour Windows File Server. Les volumes qui peuvent être configurés lors du déploiement lors de l'exécution d'une tâche, ou lors de la création ou de la mise à jour d'un service incluent Amazon EBS.

    c. Pour Type de volume, sélectionnez un type de volume compatible avec le type de configuration que vous avez sélectionné, puis configurez le type de volume.

Type de volume	Étapes	
Support Bind	a. Choisissez Add mount point (Ajouter un point de montage), puis configure z les éléments suivants : •	

Type de volume	Étapes	
	Sous Container (Conteneur), choisissez le conteneur correspondant au point de montage. • Pour Source volume (Volume source), choisissez le volume de données à monter sur le conteneur. • Sous Container path (Chemin du conteneur), saisissez le chemin d'accès au conteneur pour monter le volume. • Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume. b. Pour ajouter des points de montage supplémentaires, ajoutez un point de montage.	

Type de volume	Étapes	
EFS	a. Pour File system ID (ID du système de fichiers), choisissez l'ID du système de fichiers Amazon EFS. b. (Facultatif) Pour Root directory (Répertoire racine, entrez le répertoire du système de fichiers Amazon EFS à monter en tant que répertoire racine à l'intérieur de l'hôte. Si ce paramètre est omis, la racine du volume Amazon EFS est utilisée. Si vous prévoyez d'utiliser un point d'accès EFS, laissez ce champ vide. c. (Facultatif) Pour Access point (Point d'accès), choisissez l'ID du point d'accès à utiliser. d. (Facultatif) Pour chiffrer les données entre le système de fichiers Amazon EFS et l'hôte Amazon ECS ou pour utiliser le rôle d'exécution de tâches lors du	

Type de volume	Étapes	
	montage du volume, choisissez Advanced configurations (Configurations avancées), puis configurez les éléments suivants : • Pour chiffrer les données entre le système de fichiers Amazon EFS et l'hôte Amazon ECS, sélectionnez Transit encryption (Chiffrement de transit), puis pour Port, entrez le port à utiliser lors de l'envoi de données chiffrées entre l'hôte Amazon ECS et le serveur Amazon EFS. Si vous ne spécifiez pas de port de chiffrement en transit, il utilise la stratégie de sélection de port adoptée par l'assistant de montage Amazon EFS. Pour plus d'informations, consultez Assistant de montage EFS dans le Guide de l'utilisateur Amazon Elastic File System User.	

Type de volume	Étapes	
	• Pour utiliser le rôle IAM de tâche Amazon ECS spécifié dans une définition de tâche lors du montage du système de fichiers Amazon EFS, sélectionnez IAM authorization (Autorisation IAM). e. Choisissez Add mount point (Ajouter un point de montage), puis configurez les éléments suivants : • Sous Container (Conteneur), choisissez le conteneur correspondant au point de montage. • Pour Source volume (Volume source), choisissez le volume de données à monter sur le conteneur. • Sous Container path (Chemin du conteneur), saisissez le chemin d'accès au conteneur pour monter le volume. •	

Type de volume	Étapes	
	Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume. f. Pour ajouter des points de montage supplémen taires, ajoutez un point de montage.	

Type de volume	Étapes	
Docker	a. Pour Driver, entrez la configuration Docker du volume. Les conteneurs Windows ne prennent en charge que l'utilisation du pilote local. Pour utiliser des montages bind, spécifiez un hôte. b. Pour Scope (Portée), choisissez le cycle de vie du volume. • Pour que le cycle de vie dure au démarrage et à l'arrêt de la tâche, choisissez Task (Tâche). • Pour que le volume persiste après l'arrêt de la tâche, choisissez Shared (Partagé). c. Choisissez Add mount point (Ajouter un point de montage), puis configurez les éléments suivants : • Sous Container (Conteneur), choisissez le conteneur correspondant au point de montage.	

Type de volume	Étapes	
	• Pour Source volume (Volume source), choisissez le volume de données à monter sur le conteneur. • Sous Container path (Chemin du conteneur), saisissez le chemin d'accès au conteneur pour monter le volume. • Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume. d. Pour ajouter des points de montage supplémen taires, ajoutez un point de montage.	

Type de volume	Étapes	
FSx pour le serveur de fichiers Windows	a. Pour l'ID du système de fichiers, choisissez FSx l'ID du système de fichiers du serveur de fichiers Windows. b. Pour le répertoire racine, entrez le répertoire, entrez le répertoire dans le système FSx de fichiers Windows File Server à monter en tant que répertoire racine sur l'hôte. c. Dans Credential parameter (Paramètre d'informations d'identif ication), choisissez la manière dont les informations d'identif ication sont stockées. • Pour l'utiliser AWS Secrets Manager, entrez l'Amazon Resource Name (ARN) d'un secret Secrets Manager. • Pour l'utiliser AWS Systems Manager, entrez l'Amazon Resource Name (ARN)	

Type de volume	Étapes	
	d'un paramètre de Systems Manager. d. Pour Domaine, entrez le nom de domaine complet hébergé par un annuaire AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) ou un Active Directory auto-hébergé EC2 . e. Choisissez Add mount point (Ajouter un point de montage), puis configure z les éléments suivants : • Sous Container (Conteneur), choisisse z le conteneur correspondant au point de montage. • Pour Source volume (Volume source), choisissez le volume de données à monter sur le conteneur. • Sous Container path (Chemin du conteneur), saisissez le chemin d'accès au conteneur pour monter le volume.	

Type de volume	Étapes	
	• Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume. f. Pour ajouter des points de montage supplémen taires, ajoutez un point de montage.	

Type de volume	Étapes	
Amazon EBS	a. Choisissez Add mount point (Ajouter un point de montage), puis configure z les éléments suivants : • Sous Container (Conteneur), choisisse z le conteneur correspondant au point de montage. • Pour Source volume (Volume source), choisissez le volume de données à monter sur le conteneur. • Sous Container path (Chemin du conteneur), saisissez le chemin d'accès au conteneur pour monter le volume. • Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume. b. Pour ajouter des points de montage supplémen taires, ajoutez un point de montage.	

Type de volume	Étapes	

15. Pour ajouter un volume depuis un autre conteneur, choisissez Ajouter un volume à partir de, puis configurez les éléments suivants :
    - Pour Conteneur, choisissez le conteneur.
    - Pour Source, choisissez le conteneur contenant le volume que vous souhaitez monter.
    - Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume.
16. (Facultatif) Pour configurer les paramètres de suivi et de collecte de métriques de votre application à l'aide de AWS Distro for OpenTelemetry l'intégration, développez Monitoring, puis sélectionnez Utiliser la collecte de métriques pour collecter et envoyer des métriques pour vos tâches à Amazon CloudWatch ou à Amazon Managed Service for Prometheus. Lorsque cette option est sélectionnée, Amazon ECS crée un sidecar de AWS Distro for OpenTelemetry conteneur préconfiguré pour envoyer les métriques de l'application. Pour de plus amples informations, veuillez consulter Corrélez les performances des applications Amazon ECS à l'aide des métriques des applications.
    a. Lorsque Amazon CloudWatch est sélectionné, les métriques personnalisées de votre application sont acheminées vers CloudWatch des métriques personnalisées. Pour de plus amples informations, veuillez consulter Exporter les métriques des applications vers Amazon CloudWatch.

    > **Important**
    > Lorsque vous exportez des métriques d'application vers Amazon CloudWatch, votre définition de tâche nécessite un rôle IAM de tâche doté des autorisations requises. Pour de plus amples informations, veuillez consulter Autorisations IAM requises pour AWS Distro pour OpenTelemetry l'intégration à Amazon CloudWatch.

    b. Lorsque vous sélectionnez Amazon Managed Service for Prometheus (Prometheus libraries instrumentation) (Amazon Managed Service for Prometheus [instrumentation de bibliothèques Prometheus]), vos métriques de processeur, de la mémoire, du réseau et du stockage au niveau des tâches et vos métriques d'application personnalisées

sont acheminées vers Amazon Managed Service for Prometheus. Pour le point de terminaison d'écriture à distance de Workspace, entrez l'URL du point de terminaison d'écriture à distance de votre Prometheus espace de travail. Pour Scraping target, entrez l'hôte et le port que le AWS Distro for OpenTelemetry collecteur peut utiliser pour récupérer les données métriques. Pour de plus amples informations, veuillez consulter Exportation des métriques d'application vers Amazon Managed Service for Prometheus.

> **Important**
>
> Lorsque vous exportez des métriques d'application vers Amazon Managed Service for Prometheus, votre définition de tâche nécessite un rôle IAM de tâche avec les autorisations nécessaires. Pour de plus amples informations, veuillez consulter Autorisations IAM requises pour AWS Distro pour OpenTelemetry l'intégration à Amazon Managed Service for Prometheus.

c. Lorsque vous sélectionnez Amazon Managed Service for Prometheus OpenTelemetry (instrumentation), vos indicateurs de processeur, de mémoire, de réseau et de stockage au niveau des tâches, ainsi que les indicateurs personnalisés de votre application, sont acheminés vers Amazon Managed Service for Prometheus. Pour le point de terminaison d'écriture à distance de Workspace, entrez l'URL du point de terminaison d'écriture à distance de votre Prometheus espace de travail. Pour de plus amples informations, veuillez consulter Exportation des métriques d'application vers Amazon Managed Service for Prometheus.

> **Important**
>
> Lorsque vous exportez des métriques d'application vers Amazon Managed Service for Prometheus, votre définition de tâche nécessite un rôle IAM de tâche avec les autorisations nécessaires. Pour de plus amples informations, veuillez consulter Autorisations IAM requises pour AWS Distro pour OpenTelemetry l'intégration à Amazon Managed Service for Prometheus.

17. (Facultatif) Développez la section Tags (Identifications) pour ajouter des identifications, sous forme de paires clé-valeur, à la définition de tâche.

    - [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :
      - Pour Clé, saisissez le nom de la clé.

    - Pour Valeur, saisissez la valeur de clé.
  - [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.
18. Choisissez Créer pour enregistrer la définition de tâche.

Amazon ECS console JSON editor

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Dans le menu Créer une nouvelle définition de tâche, choisissez Créer une nouvelle définition de tâche avec JSON.
4. Dans la zone de l'éditeur JSON, modifiez votre fichier JSON,

   Le JSON doit réussir les contrôles de validation spécifiés dans the section called “Validation JSON”.
5. Choisissez Créer.

Mettre à jour une définition de tâche Amazon ECS à l'aide de la console

Une Révision de définition de tâche est une copie de la définition de tâche actuelle avec les nouvelles valeurs de paramètres remplaçant les valeurs existantes. Tous les paramètres que vous ne modifiez pas se trouvent dans la nouvelle révision.

Pour mettre à jour une définition de tâche, créez une révision de définition de tâche. Si la définition de tâche est utilisée dans un service, vous devez mettre à jour ce dernier de sorte à utiliser la définition de tâche actualisée.

Lorsque vous créez une révision, vous pouvez modifier les propriétés de conteneur et les propriétés d'environnement suivantes.

- URI de l'image de conteneur
- Mappages de port
- Variables d’environnement
- Taille de la tâche

- Taille du conteneur
- Rôle de tâche
- Rôle d'exécution de tâche
- Volumes et points de montage des conteneurs
- Registre privé

Validation JSON

L'éditeur JSON de la console Amazon ECS valide les éléments suivants dans le fichier JSON :

- Le fichier est un fichier JSON valide
- Le fichier ne contient aucune clé superflue
- Le fichier contient le paramètre `familyName`
- Il y a au moins une entrée sous `containerDefinitions`

Procédure

Amazon ECS console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, choisissez la région qui contient votre définition de tâche.
3. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
4. Choisissez la définition de tâche.
5. Sélectionnez la révision de définition de tâche, puis choisissez Créer une nouvelle révision, Créer une nouvelle révision.
6. Sur la page Create new task definition revision (Créer une nouvelle révision de définition de tâche), effectuez les modifications souhaitées. Par exemple, si vous souhaitez modifier les définitions du conteneur existantes (par exemple, l'image de conteneur, les limites de mémoire ou les mappages de ports), sélectionnez le conteneur et appliquez les modifications.
7. Vérifiez les informations, puis choisissez Mettre à jour.
8. Si la définition de tâche est utilisée dans un service, mettez à jour ce dernier avec la définition de tâche actualisée. Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.

Amazon ECS console JSON editor

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Create new revision (Créer une nouvelle révision), puis Create new revision with JSON (Créer une nouvelle révision avec JSON).
4. Dans la zone de l'éditeur JSON, modifiez votre fichier JSON,

   Le JSON doit réussir les contrôles de validation spécifiés dans the section called “Validation JSON”.
5. Choisissez Créer.

Annulation de l'enregistrement d'une révision de définition de tâche Amazon ECS à l'aide de la console

Vous pouvez annuler l'enregistrement de la révision de définition de tâche afin qu'elle ne s'affiche plus dans vos appels d'`ListTaskDefinition`API ou dans la console lorsque vous souhaitez exécuter une tâche ou mettre à jour un service.

Lorsque vous annulez l'enregistrement de la révision d'une définition de tâche, celle-ci est immédiatement marquée comme `INACTIVE`. Les tâches et services existants qui font référence à une révision de définition de tâche `INACTIVE` continuent à s'exécuter sans interruption. Les services existants qui font référence à une révision de définition de tâche `INACTIVE` peuvent toujours être mis à l'échelle en augmentant ou réduisant le nombre de services souhaité.

Vous ne pouvez pas utiliser une révision de définition de tâche `INACTIVE` pour exécuter de nouvelles tâches ou créer de nouveaux services. Vous ne pouvez pas non plus mettre à jour un service existant pour faire référence à une révision de définition de tâche `INACTIVE` (mais ces restrictions peuvent ne prendre effet que jusqu'à 10 minutes suivant l'annulation de l'enregistrement).

> **Note**
>
> Lorsque vous annulez l'enregistrement de toutes les révisions dans une famille de tâches, la famille de définition de tâche est déplacée vers la liste `INACTIVE`. L'ajout d'une nouvelle révision d'une définition de tâche `INACTIVE` ramène la famille de définition de tâche dans la liste `ACTIVE`.

À l'heure actuelle, les révisions de définition de tâche `INACTIVE` restent détectables dans votre compte indéfiniment. Cependant, ce comportement est sujet à changement à l'avenir. Par conséquent, vous ne devriez pas compter sur une conservation de révisions de définition de tâche `INACTIVE` au-delà du cycle de vie des tâches et services associés.

AWS CloudFormation piles

Le comportement suivant s'applique aux définitions de tâches créées dans la nouvelle console Amazon ECS avant le 12 janvier 2023.

Lorsque vous créez une définition de tâche, la console Amazon ECS crée automatiquement une CloudFormation pile dont le nom commence par`ECS-Console-V2-TaskDefinition-`. Si vous avez utilisé le AWS CLI ou un AWS SDK pour annuler l'enregistrement de la définition de tâche, vous devez supprimer manuellement la pile de définitions de tâches. Pour plus d'informations, consultez la section Suppression d'une pile dans le guide de AWS CloudFormation l'utilisateur.

Aucune CloudFormation pile n'est créée automatiquement pour les définitions de tâches créées après le 12 janvier 2023.

Procédure

Annuler l'enregistrement d'une nouvelle définition de tâche (console Amazon ECS)

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, choisissez la région qui contient votre définition de tâche.
3. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
4. Sur la page Task definitions (Définitions de tâche), choisissez la famille de la définition de tâche contenant une ou plusieurs révisions dont vous souhaitez annuler l'enregistrement.
5. Sur la page Nom de la définition de la tâche, sélectionnez les révisions à supprimer, puis choisissez Actions, Annuler l'enregistrement.
6. Vérifiez les informations contenues dans la fenêtre Deregister (Annuler l'enregistrement), puis choisissez Deregister (Annuler l'enregistrement) pour terminer.

Suppression d'une révision de définition de tâche Amazon ECS à l'aide de la console

Lorsque vous n'avez plus besoin d'une révision de définition de tâche spécifique dans Amazon ECS, vous pouvez supprimer la révision de définition de tâche.

Lorsque vous supprimez une révision de définition de tâche, elle passe immédiatement de l'état `INACTIVE` à l'état `DELETE_IN_PROGRESS`. Les tâches et services existants qui font référence à une révision de définition de tâche `DELETE_IN_PROGRESS` continuent à s'exécuter sans interruption.

Vous ne pouvez pas utiliser une révision de définition de tâche `DELETE_IN_PROGRESS` pour exécuter de nouvelles tâches ou créer de nouveaux services. Vous ne pouvez pas non plus mettre à jour un service existant pour référencer une révision de définition de tâche `DELETE_IN_PROGRESS`.

Lorsque vous supprimez toutes les révisions de définition de tâche `INACTIVE`, le nom de la définition de tâche n'est pas affiché dans la console et n'est pas renvoyé dans l'API. Si une révision de définition de tâche est dans `DELETE_IN_PROGRESS` cet état, le nom de la définition de tâche est affiché dans la console et renvoyé dans l'API. Le nom de la définition de tâche est conservé par Amazon ECS et la révision est incrémentée la prochaine fois que vous créerez une définition de tâche portant ce nom.

Ressources Amazon ECS pouvant bloquer une suppression

Une demande de suppression de définition de tâche ne sera pas traitée lorsque certaines ressources Amazon ECS dépendent de la révision de la définition de tâche. Les ressources suivantes peuvent empêcher la suppression d'une définition de tâche :

- Tâches autonomes Amazon ECS : la définition de la tâche est requise pour que la tâche reste saine.
- Tâches de service Amazon ECS : la définition de la tâche est requise pour que la tâche reste saine.
- Déploiements de services et ensembles de tâches Amazon ECS : la définition des tâches est requise lorsqu'un événement de dimensionnement est lancé pour un déploiement ou un ensemble de tâches Amazon ECS.

Si votre définition de tâche reste `DELETE_IN_PROGRESS` inchangée, vous pouvez utiliser la console ou le AWS CLI pour identifier, puis arrêter les ressources qui bloquent la suppression de la définition de tâche.

Suppression de la définition de tâche après la suppression de la ressource bloquée

Les règles suivantes s'appliquent une fois que vous avez supprimé les ressources bloquant la suppression de la définition de tâche :

- Tâches Amazon ECS : la suppression de la définition de tâche peut prendre jusqu'à une heure après l'arrêt de la tâche.
- Déploiements de services et ensembles de tâches Amazon ECS : la suppression de la définition de tâche peut prendre jusqu'à 24 heures après la suppression du déploiement ou de l'ensemble de tâches.

Procédure

Pour supprimer les définitions de tâche (console Amazon ECS)

Vous devez annuler l'enregistrement d'une révision de définition de tâche avant de la supprimer. Pour de plus amples informations, veuillez consulter the section called “Annulation de l'enregistrement d'une révision de définition de tâche à l'aide de la console”.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, choisissez la région qui contient votre définition de tâche.
3. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
4. Sur la page Task definitions (Définitions de tâche), choisissez la famille de la définition de tâche contenant une ou plusieurs révisions que vous souhaitez supprimer.
5. Sur la page du nom de la définition de tâche, sélectionnez les révisions à supprimer, puis choisissez Actions, Supprimer.

   Si l'option Supprimer n'est pas disponible, vous devez annuler l'enregistrement de la définition de tâche.
6. Vérifiez les informations dans le champ de confirmation Supprimer, puis choisissez Supprimer pour terminer.

Cas d'utilisation de la définition des tâches Amazon ECS

Découvrez comment rédiger des définitions de tâches pour différents AWS services et fonctionnalités.

En fonction de votre charge de travail, certains paramètres de définition des tâches doivent être définis. En ce qui concerne le type de EC2 lancement, vous devez également choisir des instances spécifiques conçues pour la charge de travail.

Rubriques

- Définitions de tâches Amazon ECS pour les charges de travail du GPU
- Définitions de tâches Amazon ECS pour les charges de travail de transcodage vidéo
- Définitions de tâches Amazon ECS pour les charges de travail d'apprentissage automatique AWS Neuron
- Définitions de tâches Amazon ECS pour les instances de deep learning
- Définitions de tâches Amazon ECS pour les charges de travail ARM 64 bits
- Envoyez les journaux Amazon ECS à CloudWatch
- Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner
- Utilisation d'images autres que des AWS conteneurs dans Amazon ECS
- Redémarrer des conteneurs individuels dans des tâches Amazon ECS avec des politiques de redémarrage de conteneurs
- Transférer des données sensibles vers un conteneur Amazon ECS

Définitions de tâches Amazon ECS pour les charges de travail du GPU

Amazon ECS prend en charge les charges de travail qui utilisent GPUs, lorsque vous créez des clusters, avec des instances de conteneur compatibles GPUs. Les instances de conteneur EC2 basées sur le GPU Amazon qui utilisent les types d'instances p2, p3, p5, g3, g4 et g5 donnent accès à NVIDIA. GPUs Pour plus d'informations, consultez la section Instances de calcul accéléré Linux dans le guide Amazon EC2 Instance Types.

Amazon ECS fournit une AMI optimisée pour le GPU qui est fournie avec des pilotes de noyau NVIDIA préconfigurés et une exécution du GPU du Docker. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.

Vous pouvez en désigner un certain nombre GPUs dans votre définition de tâche pour prendre en compte le placement des tâches au niveau du conteneur. Amazon ECS planifie les instances de conteneur disponibles qui prennent en charge GPUs et épinglent GPUs les conteneurs physiques aux conteneurs appropriés pour des performances optimales.

Les types d'instances EC2 basés sur le GPU Amazon suivants sont pris en charge. Pour plus d'informations, consultez les instances Amazon EC2 P2, les instancesAmazon EC2 P3, les instancesAmazon EC2 P4d, les instances Amazon EC2 P5,les instances Amazon EC2 G3,les instances Amazon EC2 G4, les instances Amazon EC2 G5, les instancesAmazon EC2 G6 et les instancesAmazon G6e. EC2

Type d’instance	GPUs	Mémoire GPU (Gio)	v CPUs	Mémoire (Gio)
p3.2xlarge	1	16	8	61
p3.8xlarge	4	64	32	244
p3.16xlarge	8	128	64	488
p3dn.24xlarge	8	256	96	768
p4d.24xlarge	8	320	96	1 152
p5.48xlarge	8	640	192	2048
g3s.xlarge	1	8	4	30,5
g3.4xlarge	1	8	16	122
g3.8xlarge	2	16	32	244
g3.16xlarge	4	32	64	488
g4dn.xlarge	1	16	4	16
g4dn.2xlarge	1	16	8	32
g4dn.4xlarge	1	16	16	64
g4dn.8xlarge	1	16	32	128

Type d'instance	GPUs	Mémoire GPU (Gio)	v CPUs	Mémoire (Gio)
g4dn.12xlarge	4	64	48	192
g4dn.16xlarge	1	16	64	256
g5.xlarge	1	24	4	16
g5.2xlarge	1	24	8	32
g5.4xlarge	1	24	16	64
g5.8xlarge	1	24	32	128
g5.16xlarge	1	24	64	256
g5.12xlarge	4	96	48	192
g5.24xlarge	4	96	96	384
g5.48xlarge	8	192	192	768
g6.xlarge	1	24	4	16
g 6,2 x large	1	24	8	32
g 6,4 x large	1	24	16	64
g 6,8 x large	1	24	32	128
g6.16.x large	1	24	64	256
g 6,12 x large	4	96	48	192
g 6,24 x large	4	96	96	384
g 6,48 x large	8	192	192	768
g6.metal	8	192	192	768
gr6,4 x large	1	24	16	128

Type d'instance	GPUs	Mémoire GPU (Gio)	v CPUs	Mémoire (Gio)
g6e.xlarge	1	48	4	32
g6e.2xlarge	1	48	8	64
g6e.4xlarge	1	48	16	128
6 x 8 x large	1	48	32	256
g6e16.x large	1	48	64	512
g6e12.x large	4	192	48	384
g6e24.x large	4	192	96	768
g6e48.x large	8	384	192	1536
gr6,8 x large	1	24	32	256

Vous pouvez récupérer l'identifiant Amazon Machine Image (AMI) pour Amazon ECS-Optimized AMIs en interrogeant l'API AWS Systems Manager Parameter Store. Avec ce paramètre, vous n'avez pas besoin de rechercher manuellement l'AMI optimisée pour Amazon ECS. IDs Pour plus d'informations sur l'API du magasin de paramètres de Systems Manager, consultez GetParameter. L'utilisateur que vous utilisez doit disposer de l'autorisation IAM `ssm:GetParameter` pour récupérer les métadonnées de l'AMI optimisée pour Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

Considérations

> **Note**
>
> La prise en charge du type de famille d'instances g2 est obsolète.
> Le type de famille d'instances p2 n'est pris en charge que sur les versions antérieures à la version `20230912` des AMI optimisées pour le GPU Amazon ECS. Si vous devez continuer à utiliser des instances p2, veuillez consulter Que faire si vous avez besoin d'une instance P2.

Les mises à jour sur place des NVIDIA/CUDA pilotes sur ces deux types de familles d'instances peuvent entraîner des défaillances de la charge de travail du GPU.

Nous vous recommandons de prendre en compte les points suivants avant de commencer à travailler avec GPUs Amazon ECS.

- Vos clusters peuvent contenir une combinaison d'instances de conteneur GPU et non GPU.
- Vous pouvez exécuter des charges de travail GPU sur des instances externes. Lorsque vous enregistrez une instance externe auprès de votre cluster, assurez-vous que l'indicateur `--enable-gpu` est inclus dans le script d'installation. Pour de plus amples informations, veuillez consulter Enregistrement d'une instance externe dans un cluster Amazon ECS.
- Vous devez définir `ECS_ENABLE_GPU_SUPPORT` sur `true` dans votre fichier de configuration d'agent. Pour de plus amples informations, veuillez consulter the section called “Configuration de l'agent de conteneur”.
- Lorsque vous exécutez une tâche ou créez un service, vous pouvez utiliser des attributs de type d'instance lors de la configuration des contraintes de placement des tâches afin de déterminer sur quelles instances de conteneur la tâche est lancée. Ainsi, vous pouvez utiliser plus efficacement vos ressources. Pour de plus amples informations, veuillez consulter Comment Amazon ECS place les tâches sur les instances de conteneur.

  L'exemple suivant lance une tâche sur une instance de conteneur `g4dn.xlarge` dans votre cluster par défaut.

  ```
  aws ecs run-task --cluster default --task-definition ecs-gpu-task-def \
      --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
    g4dn.xlarge" --region us-east-2
  ```

- Pour chaque conteneur dont les besoins en ressources GPU sont spécifiés dans la définition du conteneur, Amazon ECS définit l'exécution du conteneur comme étant celle du conteneur NVIDIA.
- L'exécution du conteneur NVIDIA nécessite que certaines variables d'environnement soient définies dans le conteneur pour fonctionner correctement. Pour obtenir la liste de ces variables d'environnement, consultez Configurations spécialisées avec Docker. Amazon ECS définit la valeur de la variable d'`NVIDIA_VISIBLE_DEVICES`environnement comme une liste des périphériques GPU IDs qu'Amazon ECS attribue au conteneur. Pour les autres variables d'environnement requises, Amazon ECS ne les définit pas. Assurez-vous que votre image de conteneur les définit ou qu'elles sont définies dans la définition du conteneur.

- La famille de types d'instances p5 est prise en charge sur la version `20230929` et les versions ultérieures des AMI optimisées pour le GPU Amazon ECS.
- La famille de types d'instance g4 est prise en charge sur la version `20230913` et les versions ultérieures des AMI optimisées pour le GPU Amazon ECS. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs. Elle n'est pas prise en charge dans le flux de travail Create Cluster (Créer un cluster) dans la console Amazon ECS. Pour utiliser ces types d'instances, vous devez utiliser la EC2 console Amazon ou l'API et enregistrer manuellement les instances dans votre cluster. AWS CLI
- Le type d'instance p4d.24xlarge ne fonctionne qu'avec CUDA 11 ou version ultérieure.
- L'AMI optimisée pour le GPU Amazon ECS IPv6 est activée, ce qui pose des problèmes lors de son utilisation. `yum` Ce problème peut être résolu en configurant `yum` pour l'utiliser IPv4 avec la commande suivante.

  ```
  echo "ip_resolve=4" >> /etc/yum.conf
  ```

- Lorsque vous créez une image de conteneur qui n'utilise pas les images de NVIDIA/CUDA base, vous devez définir l'une des valeurs suivantes pour la variable d'exécution du `NVIDIA_DRIVER_CAPABILITIES` conteneur :
  - `utility,compute`
  - `all`

  Pour savoir comment définir la variable, veuillez consulter Contrôle de l'exécution du conteneur NVIDIA sur le site Web de NVIDIA.
- GPUs ne sont pas pris en charge sur les conteneurs Windows.

Lancer une instance de conteneur GPU pour Amazon ECS

Pour utiliser une instance GPU sur Amazon ECS, vous devez créer un modèle de lancement, un fichier de données utilisateur, puis lancer l'instance.

Vous pouvez ensuite exécuter une tâche qui utilise une définition de tâche configurée pour le GPU.

Utiliser un modèle de lancement

Vous pouvez créer un modèle de lancement.

- Créez un modèle de lancement qui utilise l'ID d'AMI GPU optimisé pour Amazon ECS pour l'AMI. Pour plus d'informations sur la création d'un modèle de lancement, consultez Créer un

nouveau modèle de lancement à l'aide des paramètres que vous définissez dans le guide de EC2 l'utilisateur Amazon.

Utilisez l'ID AMI de l'étape précédente pour l'image Amazon Machine. Pour plus d'informations sur la façon de spécifier l'ID AMI avec le paramètre Systems Manager, consultez Spécifier un paramètre Systems Manager dans un modèle de lancement du Amazon EC2 User Guide.

Ajoutez ce qui suit aux données utilisateur du modèle de lancement. Remplacez *cluster-name* par le nom de votre cluster.

```
#!/bin/bash
echo ECS_CLUSTER=cluster-name >> /etc/ecs/ecs.config;
echo ECS_ENABLE_GPU_SUPPORT=true >> /etc/ecs/ecs.config
```

Utilisez le AWS CLI

Vous pouvez utiliser le AWS CLI pour lancer l'instance de conteneur.

1. Créez un fichier appelé `userdata.toml`. Ce fichier est utilisé pour les données utilisateur de l'instance. Remplacez *cluster-name* par le nom de votre cluster.

```
#!/bin/bash
echo ECS_CLUSTER=cluster-name >> /etc/ecs/ecs.config;
echo ECS_ENABLE_GPU_SUPPORT=true >> /etc/ecs/ecs.config
```

2. Exécutez la commande suivante pour obtenir l'ID de l'AMI du GPU. Utilisez ceci lors de l'étape suivante.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

3. Exécutez la commande suivante pour lancer l'instance GPU. N'oubliez pas de remplacer les paramètres suivants :

   - *subnet*Remplacez-le par l'ID du sous-réseau privé ou public dans lequel votre instance sera lancée.
   - *gpu_ami*Remplacez-le par l'ID AMI de l'étape précédente.
   - *t3.large*Remplacez-le par le type d'instance que vous souhaitez utiliser.
   - Remplacez *region* par le code de région.

```
aws ec2 run-instances --key-name ecs-gpu-example \
   --subnet-id subnet \
   --image-id gpu_ami \
   --instance-type t3.large \
   --region region \
   --tag-specifications 'ResourceType=instance,Tags=[{Key=GPU,Value=example}]' \
   --user-data file://userdata.toml \
   --iam-instance-profile Name=ecsInstanceRole
```

4. Exécutez la commande suivante pour vérifier que l'instance de conteneur est enregistrée auprès du cluster. Lorsque vous exécutez cette commande, n'oubliez pas de remplacer les paramètres suivants :

   - Remplacez *cluster* par le nom de votre cluster.
   - *region*Remplacez-le par votre code de région.

```
aws ecs list-container-instances --cluster cluster-name --region region
```

Spécification GPUs dans une définition de tâche Amazon ECS

Pour utiliser l'instance GPUs sur une instance de conteneur et le runtime du GPU Docker, assurez-vous d'indiquer le nombre requis par GPUs votre conteneur dans la définition de la tâche. Au fur et à mesure que les conteneurs compatibles GPUs sont placés, l'agent de conteneur Amazon ECS épingle le nombre de conteneurs physiques GPUs souhaité sur le conteneur approprié. Le nombre de conteneurs GPUs réservés à tous les conteneurs d'une tâche ne peut pas dépasser le nombre de conteneurs disponibles GPUs sur l'instance de conteneur sur laquelle la tâche est lancée. Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.

> **Important**
>
> Si vos exigences en matière de GPU ne sont pas spécifiées dans la définition de tâche, la tâche utilise l'exécution par défaut du Docker.

L'exemple suivant illustre le format JSON pour les exigences de GPU dans une définition de tâche :

```
{
  "containerDefinitions": [
     {
        ...
        "resourceRequirements" : [
            {
               "type" : "GPU",
               "value" : "2"
            }
        ],
     },
...
}
```

L'exemple suivant illustre la syntaxe d'un conteneur Docker qui spécifie une exigence GPU. Ce conteneur en utilise deux GPUs, exécute l'`nvidia-smi`utilitaire, puis se ferme.

```
{
  "containerDefinitions": [
    {
      "memory": 80,
      "essential": true,
      "name": "gpu",
      "image": "nvidia/cuda:11.0.3-base",
      "resourceRequirements": [
         {
           "type":"GPU",
           "value": "2"
         }
      ],
      "command": [
        "sh",
        "-c",
        "nvidia-smi"
      ],
      "cpu": 100
    }
  ],
  "family": "example-ecs-gpu"
}
```

Partagez GPUs

Lorsque vous souhaitez partager GPUs, vous devez configurer les éléments suivants.

1. Supprimez les exigences en matière de ressources GPU de vos définitions de tâches afin qu'Amazon ECS ne réserve aucune GPUs ressource devant être partagée.
2. Ajoutez les données utilisateur suivantes à vos instances lorsque vous souhaitez les partager GPUs. NVIDIA deviendra ainsi le moteur d'exécution du conteneur Docker par défaut sur l'instance de conteneur afin que tous les conteneurs Amazon ECS puissent utiliser le GPUs. Pour plus d'informations, consultez la section Exécuter des commandes lorsque vous lancez une EC2 instance avec saisie de données utilisateur dans le guide de EC2 l'utilisateur Amazon.

```
const userData = ec2.UserData.forLinux();
 userData.addCommands(
 'sudo rm /etc/sysconfig/docker',
 'echo DAEMON_MAXFILES=1048576 | sudo tee -a /etc/sysconfig/docker',
 'echo OPTIONS="--default-ulimit nofile=32768:65536 --default-runtime nvidia" |
 sudo tee -a /etc/sysconfig/docker',
 'echo DAEMON_PIDFILE_TIMEOUT=10 | sudo tee -a /etc/sysconfig/docker',
 'sudo systemctl restart docker',
);
```

3. Définissez la variable d'`NVIDIA_VISIBLE_DEVICES`environnement sur votre conteneur. Vous pouvez le faire en spécifiant la variable d'environnement dans votre définition de tâche. Pour plus d'informations sur les valeurs valides, consultez l'énumération des GPU sur le site de documentation de NVIDIA.

Que faire si vous avez besoin d'une instance P2

Si vous avez besoin d'utiliser une instance P2, vous pouvez utiliser l'une des options suivantes pour continuer à utiliser ces instances.

Vous devez modifier les données utilisateur de l'instance pour les deux options. Pour plus d'informations, consultez la section Exécuter des commandes lorsque vous lancez une EC2 instance avec saisie de données utilisateur dans le guide de EC2 l'utilisateur Amazon.

Utiliser la dernière AMI optimisée pour le GPU prise en charge

Vous pouvez utiliser la version `20230906` de l'AMI optimisée pour le GPU et ajouter les éléments suivants aux données utilisateur de l'instance.

Remplacez cluster-name par le nom de votre cluster.

```
#!/bin/bash
echo "exclude=*nvidia* *cuda*" >> /etc/yum.conf
echo "ECS_CLUSTER=cluster-name" >> /etc/ecs/ecs.config
```

Utiliser la dernière AMI optimisée pour le GPU et mettre à jour les données utilisateur

Vous pouvez ajouter ce qui suit aux données utilisateur de l'instance. Cela désinstalle les pilotes Nvidia 535/Cuda12.2, puis installe les pilotes Nvidia 470/Cuda11.4 et corrige la version.

```
#!/bin/bash
yum remove -y cuda-toolkit* nvidia-driver-latest-dkms*
tmpfile=$(mktemp)
cat >$tmpfile <<EOF
[amzn2-nvidia]
name=Amazon Linux 2 Nvidia repository
mirrorlist=\$awsproto://\$amazonlinux.\$awsregion.\$awsdomain/\$releasever/amzn2-
nvidia/latest/\$basearch/mirror.list
priority=20
gpgcheck=1
gpgkey=https://developer.download.nvidia.com/compute/cuda/repos/rhel7/
x86_64/7fa2af80.pub
enabled=1
exclude=libglvnd-*
EOF

mv $tmpfile /etc/yum.repos.d/amzn2-nvidia-tmp.repo
yum install -y system-release-nvidia cuda-toolkit-11-4 nvidia-driver-latest-
dkms-470.182.03
yum install -y libnvidia-container-1.4.0 libnvidia-container-tools-1.4.0 nvidia-
container-runtime-hook-1.4.0 docker-runtime-nvidia-1

echo "exclude=*nvidia* *cuda*" >> /etc/yum.conf
nvidia-smi
```

Créer votre propre AMI optimisée pour le GPU compatible avec P2

Vous pouvez créer votre propre AMI Amazon ECS personnalisée optimisée pour le GPU et compatible avec les instances P2, puis lancer des instances P2 à l'aide de l'AMI.

1. Exécutez la commande suivante pour cloner le `amazon-ecs-ami repo`.

```
git clone https://github.com/aws/amazon-ecs-ami
```

2. Définissez l'agent Amazon ECS requis et les versions de l'AMI Amazon Linux source dans `release.auto.pkrvars.hcl` ou `overrides.auto.pkrvars.hcl`.
3. Exécutez la commande suivante pour créer une EC2 AMI privée compatible P2.

   Remplacez la région par la région de l'instance.

   ```
   REGION=region make al2keplergpu
   ```

4. Utilisez l'AMI avec les données utilisateur de l'instance suivantes pour vous connecter au cluster Amazon ECS.

   Remplacez cluster-name par le nom de votre cluster.

   ```
   #!/bin/bash
   echo "ECS_CLUSTER=cluster-name" >> /etc/ecs/ecs.config
   ```

Définitions de tâches Amazon ECS pour les charges de travail de transcodage vidéo

Pour utiliser les charges de travail de transcodage vidéo sur Amazon ECS, enregistrez les instances Amazon EC2 VT1. Après avoir enregistré ces instances, vous pouvez exécuter des charges de travail de transcodage vidéo en direct et prérendues en tant que tâches sur Amazon ECS. EC2 VT1 Les instances Amazon utilisent les cartes de transcodage multimédia Xilinx U30 pour accélérer les charges de travail de transcodage vidéo en direct et pré-rendues.

> **Note**
>
> Pour obtenir des instructions sur l'exécution de charges de travail de transcodage vidéo dans des conteneurs autres qu'Amazon ECS, consultez la documentation Xilinx.

Considérations

Avant de commencer VT1 le déploiement sur Amazon ECS, tenez compte des points suivants :

- Vos clusters peuvent contenir à la fois des instances VT1 et des VT1 non-instances.
- Vous avez besoin d'une application Linux qui utilise des cartes de transcodage multimédia Xilinx U30 avec des codecs AVC (H.264) et HEVC (H.265) accélérés.

> **Important**
>
> Les applications qui utilisent d'autres codecs peuvent ne pas améliorer les performances des VT1 instances.

- Une seule tâche de transcodage peut être exécutée sur une carte U30. Chaque carte est associée à deux appareils. Vous pouvez exécuter autant de tâches de transcodage qu'il y a de cartes pour chacune de vos VT1 instances.
- Lorsque vous exécutez une tâche autonome ou créez un service, vous pouvez utiliser des attributs de type d'instance lors de la configuration des contraintes de placement des tâches. Cela garantit que la tâche est lancée sur l'instance de conteneur que vous spécifiez. Cela vous permet de vous assurer que vous utilisez vos ressources de manière efficace et que vos tâches de transcodage vidéo incombent à vos VT1 instances. Pour de plus amples informations, veuillez consulter Comment Amazon ECS place les tâches sur les instances de conteneur.

  Dans l'exemple suivant, une tâche est exécutée sur une instance `vt1.3xlarge` de votre cluster `default`.

```
aws ecs run-task \
     --cluster default \
     --task-definition vt1-3xlarge-xffmpeg-processor \
     --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
 vt1.3xlarge"
```

- Vous configurez un conteneur pour utiliser une carte U30 spécifique disponible sur l'instance de conteneur hôte. Vous pouvez effectuer cette opération en utilisant le paramètre `linuxParameters` et en spécifiant les détails de l'appareil. Pour de plus amples informations, veuillez consulter Exigences relatives à la définition de tâche.

Utilisation d'une VT1 AMI

Deux options s'offrent à vous pour exécuter une AMI sur Amazon EC2 pour les instances de conteneur Amazon ECS. La première option consiste à utiliser l'AMI officielle Xilinx sur

AWS Marketplace. La deuxième option consiste à créer votre propre AMI à partir de l'exemple du référentiel.

- Xilinx propose AMIs sur le. AWS Marketplace
- Amazon ECS fournit un exemple de référentiel que vous pouvez utiliser pour créer une AMI pour les charges de travail de transcodage vidéo. Cette AMI est fournie avec les pilotes Xilinx U30. Vous pouvez trouver le référentiel contenant les scripts Packer sur GitHub. Pour plus d'informations sur Packer, consultez la documentation Packer.

Exigences relatives à la définition de tâche

Pour exécuter des conteneurs de transcodage vidéo sur Amazon ECS, votre définition de tâche doit contenir une application de transcodage vidéo utilisant les codecs H.264/AVC et H.265/HEVC accélérés. Vous pouvez créer une image de conteneur en suivant les étapes indiquées sur le Xilinx GitHub.

La définition de tâche doit être spécifique au type d'instance. Les types d'instance sont 3xlarge, 6xlarge et 24xlarge. Vous devez configurer un conteneur pour utiliser des appareils Xilinx U30 spécifiques disponibles sur l'instance de conteneur hôte. Vous pouvez effectuer cette opération à l'aide du paramètre `linuxParameters`. Le tableau suivant détaille les cartes et SoCs les appareils spécifiques à chaque type d'instance.

Type d'instance	v CPUs	RAM (Gio)	Cartes accélératrices U30	XCU30 appareils SoC adressables	Chemins de l'appareil
vt1.3xlarge	12	24	1	2	`/dev/dri/renderD128 ,/dev/dri/renderD129`
vt1.6xlarge	24	48	2	4	`/dev/dri/renderD128 ,/dev/`

Type d'instance	v CPUs	RAM (Gio)	Cartes accélérat rices U30	XCU30 appareils SoC adressables	Chemins de l'appareil
					`dri/renderD129 ,/dev/dri/renderD130 ,/dev/dri/renderD131`

Type d'instance	v CPUs	RAM (Gio)	Cartes accélérat rices U30	XCU30 appareils SoC adressables	Chemins de l'appareil
vt1.24xlarge	96	182	8	16	`/dev/dri/renderD128 ,/dev/dri/renderD129 ,/dev/dri/renderD130 ,/dev/dri/renderD131 ,/dev/dri/renderD132 ,/dev/dri/renderD133 ,/dev/dri/renderD134 ,/dev/dri/renderD135 ,/dev/dri/renderD136 ,/dev/dri/renderD137 ,/dev/dri/`

Type d'instance	v CPUs	RAM (Gio)	Cartes accélérat rices U30	XCU30 appareils SoC adressables	Chemins de l'appareil
					`renderD138 ,/dev/dri/renderD139 ,/dev/dri/renderD140 ,/dev/dri/renderD141 ,/dev/dri/renderD142 ,/dev/dri/renderD143`

> **Important**
>
> Si la définition de tâche répertorie les appareils que l' EC2 instance ne possède pas, la tâche ne s'exécute pas. Lorsque la tâche échoue, le message d'erreur suivant s'affiche dans `stoppedReason` : `CannotStartContainerError: Error response from daemon: error gathering device information while adding custom device "/dev/dri/renderD130": no such file or directory`.

Spécification du transcodage vidéo dans une définition de tâche Amazon ECS

Dans l'exemple suivant, la syntaxe utilisée pour la définition de tâche d'un conteneur Linux sur Amazon EC2 est fournie. Cette définition de tâche concerne les images de conteneur créées

conformément à la procédure fournie dans la documentation Xilinx. Si vous utilisez cet exemple, remplacez `image` par votre propre image, et copiez vos fichiers vidéo dans l'instance dans le répertoire `/home/ec2-user`.

vt1.3xlarge

1. Créez un fichier texte nommé `vt1-3xlarge-ffmpeg-linux.json` avec le contenu suivant.

```
{
    "family": "vt1-3xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.3xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
```

```
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Enregistrez la définition de tâche.

```
aws ecs register-task-definition --family vt1-3xlarge-xffmpeg-processor --cli-
input-json file://vt1-3xlarge-xffmpeg-linux.json --region us-east-1
```

vt1.6xlarge

1. Créez un fichier texte nommé `vt1-6xlarge-ffmpeg-linux.json` avec le contenu suivant.

```
{
    "family": "vt1-6xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
```

```
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.6xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD130",
                        "hostPath": "/dev/dri/renderD130",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD131",
```

```
                        "hostPath": "/dev/dri/renderD131",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Enregistrez la définition de tâche.

```
aws ecs register-task-definition --family vt1-6xlarge-xffmpeg-processor --cli-
input-json file://vt1-6xlarge-xffmpeg-linux.json --region us-east-1
```

vt1.24xlarge

1. Créez un fichier texte nommé `vt1-24xlarge-ffmpeg-linux.json` avec le contenu suivant.

```
{
    "family": "vt1-24xlarge-xffmpeg-processor",
```

```
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.24xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD130",
                        "hostPath": "/dev/dri/renderD130",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
```

```
                        {
                            "containerPath": "/dev/dri/renderD131",
                            "hostPath": "/dev/dri/renderD131",
                            "permissions": [
                                "read",
                                "write"
                            ]
                        },
                        {
                            "containerPath": "/dev/dri/renderD132",
                            "hostPath": "/dev/dri/renderD132",
                            "permissions": [
                                "read",
                                "write"
                            ]
                        },
                        {
                            "containerPath": "/dev/dri/renderD133",
                            "hostPath": "/dev/dri/renderD133",
                            "permissions": [
                                "read",
                                "write"
                            ]
                        },
                        {
                            "containerPath": "/dev/dri/renderD134",
                            "hostPath": "/dev/dri/renderD134",
                            "permissions": [
                                "read",
                                "write"
                            ]
                        },
                        {
                            "containerPath": "/dev/dri/renderD135",
                            "hostPath": "/dev/dri/renderD135",
                            "permissions": [
                                "read",
                                "write"
                            ]
                        },
                        {
                            "containerPath": "/dev/dri/renderD136",
                            "hostPath": "/dev/dri/renderD136",
                            "permissions": [
```

```
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD137",
                    "hostPath": "/dev/dri/renderD137",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD138",
                    "hostPath": "/dev/dri/renderD138",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD139",
                    "hostPath": "/dev/dri/renderD139",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD140",
                    "hostPath": "/dev/dri/renderD140",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
                {
                    "containerPath": "/dev/dri/renderD141",
                    "hostPath": "/dev/dri/renderD141",
                    "permissions": [
                        "read",
                        "write"
                    ]
                },
```

```
                    {
                        "containerPath": "/dev/dri/renderD142",
                        "hostPath": "/dev/dri/renderD142",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD143",
                        "hostPath": "/dev/dri/renderD143",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Enregistrez la définition de tâche.

```
aws ecs register-task-definition --family vt1-24xlarge-xffmpeg-processor --cli-
input-json file://vt1-24xlarge-xffmpeg-linux.json --region us-east-1
```

Définitions de tâches Amazon ECS pour les charges de travail d'apprentissage automatique AWS Neuron

Vous pouvez enregistrer des instances Amazon EC2 Trn1, Amazon EC2 Inf1 et Amazon EC2 Inf2 dans vos clusters pour les charges de travail d'apprentissage automatique.

Les instances Amazon EC2 Trn1 sont alimentées par des puces AWS Trainium. Ces instances fournissent une formation performante et peu coûteuse pour machine learning dans le cloud. Vous pouvez former un modèle d'inférence basé sur machine learning à l'aide d'un cadre de machine learning avec AWS Neuron sur une instance Trn1. Vous pouvez ensuite exécuter le modèle sur une instance Inf1 ou sur une instance Inf2 pour utiliser l'accélération des puces AWS Inferentia.

Les instances Amazon EC2 Inf1 et Inf2 sont alimentées par des puces AWS Inferentia qui fournissent des performances élevées et une inférence à moindre coût dans le cloud.

Les modèles de machine learning sont déployés sur des conteneurs à l'aide d'AWS Neuron, qui est un kit de développement logiciel (SDK) spécialisé. Le SDK comprend un compilateur, un environnement d'exécution et des outils de profilage qui optimisent les performances d'apprentissage automatique des puces d'apprentissage AWS automatique. AWS Neuron prend en charge les frameworks d'apprentissage automatique populaires tels que TensorFlow, PyTorch, et Apache MXNet.

Considérations

Avant de commencer à déployer Neuron sur Amazon ECS, prenez en compte ce qui suit :

- Vos clusters peuvent contenir une combinaison d'instances Trn1, Inf1, Inf2 et d'autres instances.
- Vous avez besoin d'une application Linux dans un conteneur qui utilise un framework d'apprentissage automatique compatible avec AWS Neuron.

> **Important**
>
> Les applications qui utilisent d'autres cadres peuvent ne pas avoir de performances améliorées sur les instances Trn1, Inf1 et Inf2.

- Une seule tâche d'inférence ou d'entraînement d'inférence peut être exécutée sur chaque puce AWS Trainium ou AWS Inferentia. Pour Inf1, chaque puce en possède 4 NeuronCores. Pour Trn1 et Inf2, chaque puce en possède 2. NeuronCores Vous pouvez exécuter autant de tâches qu'il y a de puces pour chacune de vos instances Trn1, Inf1 et Inf2.
- Lorsque vous exécutez une tâche autonome ou créez un service, vous pouvez utiliser des attributs de type d'instance lors de la configuration des contraintes de placement des tâches. Cela garantit que la tâche est lancée sur l'instance de conteneur que vous spécifiez. Cela vous aide à optimiser l'utilisation globale des ressources et garantit que les tâches des charges de travail d'inférence sont sur vos instances Trn1, Inf1 et Inf2. Pour de plus amples informations, veuillez consulter Comment Amazon ECS place les tâches sur les instances de conteneur.

  Dans l'exemple suivant, une tâche est exécutée sur une instance `Inf1.xlarge` de votre cluster `default`.

  ```
  aws ecs run-task \
       --cluster default \
       --task-definition ecs-inference-task-def \
       --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
   Inf1.xlarge"
  ```

- Les besoins en ressources Neuron ne peuvent pas être définis dans une définition de tâche. Au lieu de cela, vous configurez un conteneur pour utiliser des puces AWS Trainium ou AWS Inferentia spécifiques disponibles sur l'instance de conteneur hôte. Pour ce faire, utilisez le paramètre `linuxParameters` et en spécifiant les détails du dispositif. Pour de plus amples informations, veuillez consulter Exigences relatives à la définition de tâche.

Utiliser l'AMI Amazon Linux 2023 (Neuron) optimisée pour Amazon ECS

Amazon ECS fournit une AMI optimisée pour Amazon ECS basée sur Amazon Linux 2023 pour les charges de travail AWS Trainium et AWS Inferentia. Il est livré avec les pilotes AWS Neuron et le runtime pour Docker. Cette AMI facilite l'exécution de charges de travail d'inférence de machine learning sur Amazon ECS.

Nous vous recommandons d'utiliser l'AMI Amazon Linux 2023 (Neuron) optimisée pour Amazon ECS lors du lancement de vos instances Amazon EC2 Trn1, Inf1 et Inf2.

Vous pouvez récupérer l'AMI Amazon Linux 2023 (Neuron) actuellement optimisée pour Amazon ECS à l'aide de la commande AWS CLI suivante.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/neuron/
recommended
```

L'AMI Amazon Linux 2023 (Neuron) optimisée pour Amazon ECS est prise en charge dans les régions suivantes :

- USA Est (Virginie du Nord)
- USA Est (Ohio)
- USA Ouest (Californie du Nord)
- USA Ouest (Oregon)
- Asie-Pacifique (Mumbai)
- Asie-Pacifique (Osaka)
- Asie-Pacifique (Séoul)
- Asie-Pacifique (Tokyo)
- Asie-Pacifique (Singapour)
- Asie-Pacifique (Sydney)
- Canada (Centre)
- Europe (Francfort)
- Europe (Irlande)
- Europe (Londres)
- Europe (Paris)
- Europe (Stockholm)
- Amérique du Sud (São Paulo)

Utiliser l'AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS

Amazon ECS fournit une AMI optimisée pour Amazon ECS basée sur Amazon Linux 2 pour les charges de travail AWS Trainium et AWS Inferentia. Il est livré avec les pilotes AWS Neuron et le runtime pour Docker. Cette AMI facilite l'exécution de charges de travail d'inférence de machine learning sur Amazon ECS.

Nous vous recommandons d'utiliser l'AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS lors du lancement de vos instances Amazon EC2 Trn1, Inf1 et Inf2.

Vous pouvez récupérer l'AMI Amazon Linux 2 (Neuron) actuellement optimisée pour Amazon ECS à l' AWS CLI aide de la commande suivante.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/inf/
recommended
```

L'AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS est prise en charge dans les régions suivantes :

- USA Est (Virginie du Nord)
- USA Est (Ohio)
- USA Ouest (Californie du Nord)
- USA Ouest (Oregon)
- Asie-Pacifique (Mumbai)
- Asie-Pacifique (Osaka)
- Asie-Pacifique (Séoul)
- Asie-Pacifique (Tokyo)
- Asie-Pacifique (Singapour)
- Asie-Pacifique (Sydney)
- Canada (Centre)
- Europe (Francfort)
- Europe (Irlande)
- Europe (Londres)
- Europe (Paris)
- Europe (Stockholm)
- Amérique du Sud (São Paulo)

Exigences relatives à la définition de tâche

Pour déployer Neuron sur Amazon ECS, votre définition de tâche doit contenir la définition du conteneur d'un conteneur prédéfini servant le modèle d'inférence pour. TensorFlow Il est fourni par AWS Deep Learning Containers. Ce conteneur contient le runtime AWS Neuron et l'application TensorFlow Serving. Au démarrage, ce conteneur récupère votre modèle depuis Amazon S3, lance Neuron TensorFlow Serving avec le modèle enregistré et attend les demandes de prédiction. Dans

l'exemple suivant, l'image du conteneur est TensorFlow 1.15 et Ubuntu 18.04. Une liste complète des Deep Learning Containers prédéfinis optimisés pour Neuron est disponible sur. GitHub Pour plus d'informations, consultez la section Utilisation de AWS Neuron TensorFlow Serving.

```
763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-inference-neuron:1.15.4-neuron-
py37-ubuntu18.04
```

Alternativement, vous pouvez également créer votre propre image de conteneur sidecar Neuron. Pour plus d'informations, consultez Tutoriel : Neuron TensorFlow Serving dans le guide du AWS Apprentissage profond (deep learning) AMIs développeur.

La définition de tâche doit être spécifique au type d'instance unique. Vous devez configurer un conteneur pour utiliser des appareils AWS Trainium ou AWS Inferentia spécifiques disponibles sur l'instance de conteneur hôte. Vous pouvez effectuer cette opération à l'aide du paramètre `linuxParameters`. Le tableau suivant détaille les cartes et les puces qui sont spécifiques à chaque type d'instance.

Type d'instance	v CPUs	RAM (Gio)	AWS puces accélératrices ML	Chemins de l'appareil
trn1.2xlarge	8	32	1	`/dev/neuron0`
trn1.32xlarge	128	512	16	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/`

Type d'instance	v CPUs	RAM (Gio)	AWS puces accélératrices ML	Chemins de l'appareil
				`neuron10` , `/dev/neuron11` , `/dev/neuron12` , `/dev/neuron13` , `/dev/neuron14` , `/dev/neuron15`
inf1.xlarge	4	8	1	`/dev/neuron0`
inf1.2xlarge	8	16	1	`/dev/neuron0`
inf1.6xlarge	24	48	4	`/dev/neuron0` , `/dev/neuron1` , `/dev/neuron2` , `/dev/neuron3`

Type d'instance	v CPUs	RAM (Gio)	AWS puces accélératrices ML	Chemins de l'appareil
inf1.24xlarge	96	192	16	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/neuron10 ,/dev/neuron11 ,/dev/neuron12 ,/dev/neuron13 ,/dev/neuron14 ,/dev/neuron15`
inf2.xlarge	8	16	1	`/dev/neuron0`
inf2.8xlarge	32	64	1	`/dev/neuron0`

Type d'instance	v CPUs	RAM (Gio)	AWS puces accélératrices ML	Chemins de l'appareil
inf2.24xlarge	96	384	6	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,`
inf2.48xlarge	192	768	12	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/neuron10 ,/dev/neuron11`

Spécification de l'apprentissage automatique AWS Neuron dans une définition de tâche Amazon ECS

Voici un exemple de définition de tâche Linux pour `inf1.xlarge` qui affiche la syntaxe à utiliser.

```
{
    "family": "ecs-neuron",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == inf1.xlarge"
        }
    ],
    "executionRoleArn": "${YOUR_EXECUTION_ROLE}",
    "containerDefinitions": [
        {
            "entryPoint": [
                "/usr/local/bin/entrypoint.sh",
                "--port=8500",
                "--rest_api_port=9000",
                "--model_name=resnet50_neuron",
                "--model_base_path=s3://amzn-s3-demo-bucket/resnet50_neuron/"
            ],
            "portMappings": [
                {
                    "hostPort": 8500,
                    "protocol": "tcp",
                    "containerPort": 8500
                },
                {
                    "hostPort": 8501,
                    "protocol": "tcp",
                    "containerPort": 8501
                },
                {
                    "hostPort": 0,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "linuxParameters": {
                "devices": [
                    {
```

```
                        "containerPath": "/dev/neuron0",
                        "hostPath": "/dev/neuron0",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ],
                "capabilities": {
                    "add": [
                        "IPC_LOCK"
                    ]
                }
            },
            "cpu": 0,
            "memoryReservation": 1000,
            "image": "763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-
inference-neuron:1.15.4-neuron-py37-ubuntu18.04",
            "essential": true,
            "name": "resnet50"
        }
    ]
}
```

Définitions de tâches Amazon ECS pour les instances de deep learning

Pour utiliser les charges de travail d'apprentissage profond sur Amazon ECS, enregistrez EC2 DL1 les instances Amazon dans vos clusters. Les EC2 DL1 instances Amazon sont alimentées par les accélérateurs Gaudi de Habana Labs (une société Intel). Utilisez le SDK Habana SynapseAI pour vous connecter aux accélérateurs Habana Gaudi. Le SDK prend en charge les frameworks d'apprentissage automatique populaires, TensorFlow et PyTorch.

Considérations

Avant de commencer DL1 le déploiement sur Amazon ECS, tenez compte des points suivants :

- Vos clusters peuvent contenir à la fois des instances DL1 et des DL1 non-instances.
- Lorsque vous exécutez une tâche autonome ou créez un service, vous pouvez utiliser des attributs de type d'instance lors de la configuration des contraintes de placement des tâches afin de garantir sur quelles instances de conteneur la tâche, que vous spécifiez, est lancée. Cela garantit que vos ressources sont utilisées efficacement et que les tâches relatives aux charges de travail liées

au deep learning se situent sur vos DL1 instances. Pour de plus amples informations, veuillez consulter Comment Amazon ECS place les tâches sur les instances de conteneur.

L'exemple suivant exécute une tâche sur une instance `dl1.24xlarge` de votre cluster `default`.

```
aws ecs run-task \
     --cluster default \
     --task-definition ecs-dl1-task-def \
     --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
 dl1.24xlarge"
```

Utilisation d'une DL1 AMI

Trois options s'offrent à vous pour exécuter une AMI sur EC2 DL1 des instances Amazon pour Amazon ECS :

- AWS Marketplace AMIs qui sont fournis par Habana ici.
- Habana Deep Learning AMIs fourni par Amazon Web Services. Comme il n'est pas inclus, vous devez installer l'agent de conteneur Amazon ECS séparément.
- Utilisez Packer pour créer une AMI personnalisée fournie par le GitHubdépôt. Pour plus d'informations, consultez la section documentation Packer.

Spécifier le deep learning dans une définition de tâche Amazon ECS

Pour exécuter des conteneurs d'apprentissage profond accéléré Habana Gaudi sur Amazon ECS, votre définition de tâche doit contenir la définition de conteneur d'un conteneur prédéfini qui sert le modèle d'apprentissage profond pour TensorFlow ou PyTorch utilisant Habana SynapseAI fourni par Deep Learning Containers. AWS

L'image de conteneur suivante contient la TensorFlow version 2.7.0 et Ubuntu 20.04. Une liste complète des Deep Learning Containers prédéfinis et optimisés pour les accélérateurs Habana Gaudi est mise à jour sur. GitHub Pour de plus amples informations, veuillez consulter la section Conteneurs d'entraînement Habana.

```
763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training-habana:2.7.0-hpu-py38-
synapseai1.2.0-ubuntu20.04
```

Voici un exemple de définition de tâche pour les conteneurs Linux sur Amazon EC2, avec la syntaxe à utiliser. Cet exemple utilise une image contenant l'outil HL-SMI (HL-SMI) Habana Labs, disponible ici : `vault.habana.ai/gaudi-docker/1.1.0/ubuntu20.04/habanalabs/tensorflow-installer-tf-cpu-2.6.0:1.1.0-614`

```
{
    "family": "dl-test",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == dl1.24xlarge"
        }
    ],
    "networkMode": "host",
    "cpu": "10240",
    "memory": "1024",
    "containerDefinitions": [
        {
            "entryPoint": [
                "sh",
                "-c"
            ],
            "command": ["hl-smi"],
            "cpu": 8192,
            "environment": [
                {
                    "name": "HABANA_VISIBLE_DEVICES",
                    "value": "all"
                }
            ],
            "image": "vault.habana.ai/gaudi-docker/1.1.0/ubuntu20.04/habanalabs/
tensorflow-installer-tf-cpu-2.6.0:1.1.0-614",
            "essential": true,
            "name": "tensorflow-installer-tf-hpu"
        }
    ]
}
```

Définitions de tâches Amazon ECS pour les charges de travail ARM 64 bits

Amazon ECS prend en charge l'utilisation d'applications ARM 64 bits. Vous pouvez exécuter vos applications sur la plate-forme alimentée par les processeurs AWS Graviton. Cette plateforme est adaptée à une grande variété de charges de travail. Cela inclut les charges de travail telles que les serveurs d'applications, les microservices, le calcul hautes performances, l'inférence de machine learning basée sur le processeur, l'encodage vidéo, l'automatisation de la conception électronique, les jeux vidéos, les bases de données open source et les caches en mémoire.

Considérations

Avant de commencer à déployer des définitions de tâche utilisant l'architecture ARM 64 bits, tenez compte des points suivants :

- Les applications peuvent utiliser les types Fargate ou Launch. EC2
- Les applications ne peuvent utiliser que le système d'exploitation Linux.
- Pour le type Fargate, les applications doivent utiliser la version de plateforme `1.4.0` Fargate ou version ultérieure.
- Les applications peuvent être utilisées Fluent Bit ou à CloudWatch des fins de surveillance.
- Pour le type de lancement Fargate, les modèles Régions AWS suivants ne prennent pas en charge les charges de travail ARM 64 bits :
  - USA Est (Virginie du Nord), la zone de disponibilité `use1-az3`
- Pour le type de EC2 lancement Amazon, consultez ce qui suit pour vérifier que la région dans laquelle vous vous trouvez prend en charge le type d'instance que vous souhaitez utiliser :
  - Instances Amazon EC2 M6g
  - Instances Amazon EC2 T4g
  - Instances Amazon EC2 C6g
  - Instances Amazon EC2 R6gd
  - Instances Amazon EC2 X2GD

  Vous pouvez également utiliser la EC2 `describe-instance-type-offerings` commande Amazon avec un filtre pour afficher l'offre d'instances pour votre région.

```
aws ec2 describe-instance-type-offerings --filters Name=instance-
type,Values=instance-type --region region
```

L'exemple suivant vérifie la disponibilité du type d'instance M6 dans la Région USA Est (Virginie du Nord) (us-east-1).

```
aws ec2 describe-instance-type-offerings --filters "Name=instance-type,Values=m6*" --
region us-east-1
```

Pour plus d'informations, consultez describe-instance-type-offerings le manuel Amazon EC2 Command Line Reference.

Spécification de l'architecture ARM dans une définition de tâche Amazon ECS

Pour utiliser l'architecture ARM, spécifiez `ARM64` pour paramètre de la définition de la tâche `cpuArchitecture`.

Dans l'exemple suivant, l'architecture ARM est spécifiée dans une définition de tâche. Elles sont au format JSON.

```
{
    "runtimePlatform": {
        "operatingSystemFamily": "LINUX",
        "cpuArchitecture": "ARM64"
    },
...
}
```

Dans l'exemple suivant, une définition de tâche pour l'architecture ARM affiche « hello world ».

```
{
 "family": "arm64-testapp",
 "networkMode": "awsvpc",
 "containerDefinitions": [
    {
        "name": "arm-container",
        "image": "public.ecr.aws/docker/library/busybox:latest",
        "cpu": 100,
        "memory": 100,
        "essential": true,
        "command": [ "echo hello world" ],
        "entryPoint": [ "sh", "-c" ]
    }
```

```
 ],
 "requiresCompatibilities": [ "EC2" ],
 "cpu": "256",
 "memory": "512",
 "runtimePlatform": {
        "operatingSystemFamily": "LINUX",
        "cpuArchitecture": "ARM64"
  },
 "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole"
}
```

Envoyez les journaux Amazon ECS à CloudWatch

Vous pouvez configurer les conteneurs de vos tâches pour envoyer des informations de journal à CloudWatch Logs. Si vous utilisez le type de lancement Fargate pour vos tâches, vous pouvez afficher les journaux de vos conteneurs. Si vous utilisez le type de EC2 lancement, vous pouvez consulter les différents journaux de vos conteneurs à un seul endroit pratique, et cela évite que vos journaux de conteneurs n'occupent de l'espace disque sur vos instances de conteneur.

> **Note**
>
> Le type des informations qui sont consignées par les conteneurs dans votre tâche dépend principalement de leur commande `ENTRYPOINT`. Par défaut, les journaux qui sont capturés affichent la sortie de la commande qui peuvent s'afficher normalement dans un terminal interactif si vous aviez exécuté le conteneur localement, c'est-à-dire les flux d'I/O `STDOUT` et `STDERR`. Le pilote de `awslogs` journal transmet simplement ces journaux de Docker à CloudWatch Logs. Pour plus d'informations sur la façon dont les journaux Docker sont traités, et notamment sur les autres façons de capturer différentes données de fichiers ou différents flux, consultez View logs for a container or service dans la documentation Docker.

Pour envoyer les journaux système de vos instances de conteneur Amazon ECS vers CloudWatch Logs, consultez la section Surveillance des fichiers journaux et CloudWatch des quotas de journaux dans le guide de l'utilisateur Amazon CloudWatch Logs.

Type de lancement Fargate

Si vous utilisez le type de lancement Fargate pour vos tâches, vous devez ajouter les paramètres `logConfiguration` requis à votre définition de tâche pour activer le pilote de journal `awslogs`.

Pour de plus amples informations, veuillez consulter Exemple de définition de tâche Amazon ECS : acheminer les journaux vers CloudWatch.

Pour le conteneur Windows sur Fargate, effectuez l'une des options suivantes lorsque l'un des paramètres de définition de tâche comporte des caractères spéciaux tels que : `& \ < > ^ |`

- Ajoutez un escape (\) avec des guillemets autour de la chaîne de paramètres entière

  exemple

  ```
  "awslogs-multiline-pattern": "\"^[|DEBUG|INFO|WARNING|ERROR\"",
  ```

- Ajoutez un caractère escape (^) autour de chaque caractère spécial

  exemple

  ```
  "awslogs-multiline-pattern": "^^[^|DEBUG^|INFO^|WARNING^|ERROR",
  ```

EC2 type de lancement

Si vous utilisez le type de EC2 lancement pour vos tâches et que vous souhaitez activer le pilote de `awslogs` journal, vos instances de conteneur Amazon ECS nécessitent au moins la version 1.9.0 de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

> **Note**
>
> Vous devez utiliser une AMI optimisée pour Amazon ECS ou une AMI personnalisée avec au moins une version `1.9.0-1` du `ecs-init` package. Lorsque vous utilisez une AMI personnalisée, vous devez spécifier que le pilote de `awslogs` journalisation est disponible sur l' EC2 instance Amazon lorsque vous démarrez l'agent en utilisant la variable d'environnement suivante dans votre docker run instruction ou votre fichier de variable d'environnement.
>
> ```
> ECS_AVAILABLE_LOGGING_DRIVERS=["json-file","awslogs"]
> ```

Vos instances de conteneur Amazon ECS nécessitent également une autorisation `logs:CreateLogStream` et `logs:PutLogEvents` sur le rôle IAM avec lequel vous pouvez lancer vos instances de conteneur. Si vous avez créé votre rôle d'instance de conteneur Amazon ECS avant l'activation de la prise en charge du pilote du journal `awslogs` dans Amazon ECS, vous devrez sans doute ajouter ces autorisations. `ecsTaskExecutionRole` est utilisé lorsqu'il est affecté à la tâche et contient probablement les autorisations appropriées. Pour plus d'informations sur le rôle d'exécution des tâches, consultezRôle IAM d'exécution de tâche Amazon ECS. Si vos instances de conteneur utilisent la politique IAM gérée pour les instances de conteneur, vos instances de conteneur disposent probablement des autorisations appropriées. Pour plus d'informations sur la politique IAM gérée pour les instances de conteneur, consultezRôle IAM d'instance de conteneur Amazon ECS.

Exemple de définition de tâche Amazon ECS : acheminer les journaux vers CloudWatch

Avant que vos conteneurs puissent envoyer des journaux CloudWatch, vous devez spécifier le pilote de `awslogs` journal pour les conteneurs dans votre définition de tâche. Pour plus d'informations sur les paramètres du journal, voir Stockage et journalisation

La définition de tâche JSON qui suit possède un objet `logConfiguration` spécifié pour chaque conteneur. L'un concerne le WordPress conteneur qui envoie les journaux à un groupe de journaux appelé`awslogs-wordpress`. L'autre concerne un conteneur MySQL qui envoie des journaux à un groupe de journaux appelé `awslogs-mysql`. Les deux conteneurs utilisent le préfixe de flux de journal `awslogs-example`.

```
{
    "containerDefinitions": [
        {
            "name": "wordpress",
            "links": [
                "mysql"
            ],
            "image": "public.ecr.aws/docker/library/wordpress:latest",
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80
                }
            ],
```

```
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-create-group": "true",
                    "awslogs-group": "awslogs-wordpress",
                    "awslogs-region": "us-west-2",
                    "awslogs-stream-prefix": "awslogs-example"
                }
            },
            "memory": 500,
            "cpu": 10
        },
        {
            "environment": [
                {
                    "name": "MYSQL_ROOT_PASSWORD",
                    "value": "password"
                }
            ],
            "name": "mysql",
            "image": "public.ecr.aws/docker/library/mysql:latest",
            "cpu": 10,
            "memory": 500,
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-create-group": "true",
                    "awslogs-group": "awslogs-mysql",
                    "awslogs-region": "us-west-2",
                    "awslogs-stream-prefix": "awslogs-example",
                    "mode": "non-blocking",
                    "max-buffer-size": "25m"
                }
            }
        }
    ],
    "family": "awslogs-example"
}
```

Étapes suivantes

- Vous pouvez éventuellement définir une politique de rétention pour le groupe de journaux à l'aide de l'API CloudWatch AWS CLI or. Pour plus d'informations, consultez put-retention-policy dans la référence AWS Command Line Interface .
- Après avoir enregistré une définition de tâche avec le pilote de `awslogs` journal dans une configuration de journal de définition de conteneur, vous pouvez exécuter une tâche ou créer un service avec cette définition de tâche pour commencer à envoyer des CloudWatch journaux à Logs. Pour plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS et Création d'un déploiement de mises à jour continues Amazon ECS.

Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner

Vous pouvez utiliser Amazon ECS FireLens pour utiliser les paramètres de définition des tâches pour acheminer les journaux vers un AWS service ou une destination AWS Partner Network (APN) à des fins de stockage et d'analyse des journaux. AWS Partner Network Il s'agit d'une communauté mondiale de partenaires qui tire parti des programmes, de l'expertise et des ressources pour créer, commercialiser et vendre des offres aux clients. Pour plus d'informations, veuillez consulter AWS Partner. FireLens fonctionne avec Fluentd et Fluent Bit. Nous fournissons l'image AWS pour Fluent Bit ou vous pouvez utiliser votre propre image Fluentd ou Fluent Bit.

Par défaut, Amazon ECS configure la dépendance du conteneur afin que le conteneur Firelens démarre avant tout conteneur qui l'utilise. Le conteneur Firelens s'arrête également une fois que tous les conteneurs qui l'utilisent s'arrêtent.

Tenez compte des points suivants lors de l'utilisation FireLens pour Amazon ECS :

- Nous vous recommandons de l'`my_service_`ajouter au nom du conteneur du journal afin de pouvoir facilement le distinguer dans la console.
- Amazon ECS ajoute par défaut une dépendance relative à l'ordre des conteneurs de démarrage entre les conteneurs d'applications et le FireLens conteneur. Lorsque vous spécifiez un ordre de conteneur entre les conteneurs d'applications et le FireLens conteneur, l'ordre de conteneur de départ par défaut est remplacé.
- FireLenspour Amazon ECS est pris en charge pour les tâches hébergées à la fois AWS Fargate sur Linux et EC2 sur Amazon on Linux. Les conteneurs Windows ne prennent pas en charge FireLens.

Pour plus d'informations sur la configuration de la journalisation centralisée pour les conteneurs Windows, consultez la section Journalisation centralisée pour les conteneurs Windows sur Amazon ECS à l'aide de Fluent Bit.

- Vous pouvez utiliser AWS CloudFormation des modèles FireLens pour configurer Amazon ECS. Pour plus d'informations, voir AWS::ECS::TaskDefinition FirelensConfigurationle guide de AWS CloudFormation l'utilisateur
- FireLensécoute sur le port`24224`. Pour vous assurer que le routeur de FireLens journaux n'est pas accessible en dehors de la tâche, vous ne devez pas autoriser le trafic entrant sur le port `24224` du groupe de sécurité utilisé par votre tâche. Pour les tâches utilisant le mode réseau `awsvpc`, il s'agit du groupe de sécurité associé à la tâche. Pour les tâches utilisant le mode `host` réseau, il s'agit du groupe de sécurité associé à l' EC2 instance Amazon hébergeant la tâche. Pour les tâches utilisant le mode réseau `bridge`, ne créez pas de mappages de ports utilisant le port `24224`.
- Pour les tâches qui utilisent le mode `bridge` réseau, le conteneur contenant la FireLens configuration doit démarrer avant que les conteneurs d'applications qui en dépendent ne démarrent. Pour contrôler l'ordre de début de vos conteneurs, utilisez les conditions de dépendance dans la définition de tâche. Pour de plus amples informations, veuillez consulter Dépendances du conteneur.

> **Note**
>
> Si vous utilisez des paramètres de condition de dépendance dans les définitions de conteneur avec une FireLens configuration, assurez-vous que chaque conteneur possède une exigence de `HEALTHY` condition `START` or.

- Par défaut, FireLens ajoute le nom du cluster et de la définition de tâche ainsi que le nom de ressource Amazon (ARN) du cluster en tant que clés de métadonnées stdout/stderr dans vos journaux de conteneurs. Voici un exemple du format de métadonnées.

```
"ecs_cluster": "cluster-name",
"ecs_task_arn": "arn:aws:ecs:region:111122223333:task/cluster-
name/f2ad7dba413f45ddb4EXAMPLE",
"ecs_task_definition": "task-def-name:revision",
```

Si vous ne souhaitez pas que les métadonnées figurent dans vos journaux, définissez `enable-ecs-log-metadata` sur `false`dans la section `firelensConfiguration` de la définition de tâche.

```
"firelensConfiguration":{
   "type":"fluentbit",
   "options":{
      "enable-ecs-log-metadata":"false",
      "config-file-type":"file",
      "config-file-value":"/extra.conf"
}
```

Pour utiliser cette fonctionnalité, vous devez créer un rôle IAM pour vos tâches qui fournit les autorisations nécessaires pour utiliser les AWS services requis par les tâches. Par exemple, si un conteneur achemine des journaux vers Firehose, la tâche nécessite l'autorisation d'appeler l'`firehose:PutRecordBatch`API. Pour plus d'informations, consultez Ajout et suppression d'autorisations basées sur l'identité IAM dans le Guide de l'utilisateur IAM.

Votre tâche peut également nécessiter le rôle d'exécution de tâche Amazon ECS dans les conditions suivantes. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

- Si votre tâche est hébergée sur Fargate et que vous extrayez des images de conteneurs depuis Amazon ECR ou que vous faites référence à des données sensibles AWS Secrets Manager depuis votre configuration de journal, vous devez inclure le rôle IAM d'exécution de la tâche.
- Lorsque vous utilisez un fichier de configuration personnalisé hébergé dans Amazon S3, votre rôle IAM d'exécution de tâches doit inclure l'`s3:GetObject`autorisation.

Pour plus d'informations sur l'utilisation de plusieurs fichiers de configuration avec Amazon ECS, y compris les fichiers que vous hébergez ou les fichiers dans Amazon S3, consultez Processus d'initialisation pour Fluent Bit on ECS, support multi-configurations.

Pour plus d'informations sur les exemples de configurations, voir Exemple de définition de tâche Amazon ECS : acheminer les journaux vers FireLens

Configuration des journaux Amazon ECS pour un débit élevé

Lorsque vous créez une définition de tâche, vous pouvez spécifier le nombre de lignes de journal mises en mémoire tampon en spécifiant la valeur dans le`log-driver-buffer-limit`. Pour plus d'informations, consultez Fluentd logging drive (Pilote de journalisation) dans la documentation Docker.

Utilisez cette option lorsque le débit est élevé, car Docker risque de manquer de mémoire tampon et de supprimer les messages de la mémoire tampon. Il peut donc ajouter de nouveaux messages.

Tenez compte des points suivants lors de l'utilisation FireLens pour Amazon ECS avec l'option de limite de mémoire tampon :

- Cette option est prise en charge sur le type de EC2 lancement Amazon et le type de lancement Fargate avec une version de plateforme `1.4.0` ou ultérieure.
- L'option n'est valide que lorsque `logDriver` est défini sur `awsfirelens`.
- La limite de mémoire tampon par défaut est le nombre de lignes de `1048576` journal.
- La limite de mémoire tampon doit être supérieure ou égale `0` et inférieure aux lignes de `536870912` log.
- La quantité maximale de mémoire utilisée pour cette mémoire tampon est le produit de la taille de chaque ligne de journal par la taille de la mémoire tampon. Par exemple, si les lignes de journal de l'application sont en moyenne en 2 KiB, une limite de mémoire tampon de 4 096 utiliserait au maximum 1 8 MiB. La quantité totale de mémoire allouée au niveau de la tâche doit être supérieure à la quantité de mémoire allouée à tous les conteneurs en plus de la mémoire tampon du pilote de journal.

Lorsque le pilote de journal `awsfirelens` est spécifié dans une définition de tâche, l'agent de conteneur Amazon ECS injecte les variables d'environnement suivantes dans le conteneur :

`FLUENT_HOST`

Adresse IP attribuée au FireLens conteneur.

> **Note**
>
> Si vous utilisez le type de EC2 lancement avec le mode `bridge` réseau, la variable d'`FLUENT_HOST`environnement de votre conteneur d'applications peut devenir inexacte après le redémarrage du conteneur FireLens log router (le conteneur contenant l'`firelensConfiguration`objet dans sa définition de conteneur). Cela `FLUENT_HOST` est dû au fait qu'il s'agit d'une adresse IP dynamique qui peut changer après un redémarrage. La journalisation directe depuis le conteneur de l'application vers l'adresse `FLUENT_HOST` IP peut commencer à échouer après le changement d'adresse. Pour plus d'informations sur le redémarrage de conteneurs individuels, consultezRedémarrer des

conteneurs individuels dans des tâches Amazon ECS avec des politiques de redémarrage de conteneurs.

`FLUENT_PORT`

Port sur lequel le protocole Fluent Forward écoute.

Vous pouvez utiliser les variables d'environnement `FLUENT_HOST` et `FLUENT_PORT` pour vous connecter directement au routeur de journal à partir du code au lieu de passer par `stdout`. Pour plus d'informations, voir fluent-logger-golangci-dessous GitHub.

Ce qui suit montre la syntaxe permettant de spécifier le`log-driver-buffer-limit`. Remplacez `my_service_` par le nom de votre service :

```
{
    "containerDefinitions": [
        {
            "name": "my_service_log_router",
            "image": "public.ecr.aws/aws-observability/aws-for-fluent-bit:stable",
            "cpu": 0,
            "memoryReservation": 51,
            "portMappings": [],
            "essential": true,
            "environment": [],
            "mountPoints": [],
            "volumesFrom": [],
            "user": "0",
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/ecs-aws-firelens-sidecar-container",
                    "mode": "non-blocking",
                    "awslogs-create-group": "true",
                    "max-buffer-size": "25m",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "firelens"
                },
                "secretOptions": []
            },
            "systemControls": [],
            "firelensConfiguration": {
```

```
                "type": "fluentbit"
            }
        },
        {
            "essential": true,
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "firehose",
                    "region": "us-west-2",
                    "delivery_stream": "my-stream",
                    "log-driver-buffer-limit": "51200"
                }
            },
            "dependsOn": [
                {
                    "containerName": "log_router",
                    "condition": "START"
                }
            ],
            "memoryReservation": 100
        }
    ]
}
```

AWS pour les référentiels Fluent Bit d'images pour Amazon ECS

AWS fournit une Fluent Bit image avec des plugins pour CloudWatch Logs et Firehose. Nous vous recommandons d'utiliser Fluent Bit comme routeur de journal car il a un taux d'utilisation des ressources inférieur à Fluentd. Pour plus d'informations, consultez CloudWatch Logs for Fluent Bit et Amazon Kinesis Firehose for Fluent Bit.

L'image AWS for Fluent Bit est disponible sur Amazon ECR à la fois dans la galerie publique Amazon ECR et dans un référentiel Amazon ECR dans la plupart des cas Régions AWS pour une haute disponibilité.

Galerie publique Amazon ECR

L'Fluent Bitimage AWS pour est disponible sur la galerie publique Amazon ECR. Il s'agit de l'emplacement recommandé pour télécharger l'Fluent Bitimage AWS for, car il s'agit d'un dépôt public

et peut être utilisé par tous Régions AWS. Pour plus d'informations, consultez aws-for-fluent-bitla galerie publique Amazon ECR.

Linux

L'Fluent Bitimage AWS for de la galerie publique Amazon ECR prend en charge le système d'exploitation Amazon Linux avec l'`ARM 64`x`86-64`architecture or.

Vous pouvez extraire l'Fluent Bitimage AWS for de la galerie publique Amazon ECR en spécifiant l'URL du référentiel avec la balise d'image souhaitée. Les étiquettes d'image disponibles peuvent être trouvées sur la page Étiquettes d'image dans la galerie publique Amazon ECR.

L'exemple suivant montre la syntaxe à utiliser pour la CLI Docker.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:tag
```

Par exemple, vous pouvez extraire la dernière version stable AWS pour l'Fluent Bitimage à l'aide de cette commande Docker CLI.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:stable
```

> **Note**
>
> Les extractions non authentifiées sont autorisées, mais ont une limite de débit inférieure à celle des extractions authentifiées. Pour vous authentifier à l'aide de votre AWS compte avant de le tirer, utilisez la commande suivante.
>
> ```
> aws ecr-public get-login-password --region us-east-1 | docker login --username
> AWS --password-stdin public.ecr.aws
> ```

Windows

L'Fluent Bitimage AWS for de la galerie publique Amazon ECR prend en charge l'`AMD64`architecture avec les systèmes d'exploitation suivants :

- Windows Server 2022 Full
- Windows Server 2022 Core
- Windows Server 2019 Full

- Windows Server 2019 Core

Les conteneurs Windows installés sur AWS Fargate ne sont pas pris en charge. FireLens

Vous pouvez extraire l'Fluent Bitimage AWS for de la galerie publique Amazon ECR en spécifiant l'URL du référentiel avec la balise d'image souhaitée. Les étiquettes d'image disponibles peuvent être trouvées sur la page Étiquettes d'image dans la galerie publique Amazon ECR.

L'exemple suivant montre la syntaxe à utiliser pour la CLI Docker.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:tag
```

Par exemple, vous pouvez extraire l'Fluent Bitimage stable la plus récente AWS à l'aide de cette commande Docker CLI.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:windowsservercore-
stable
```

> **Note**
>
> Les extractions non authentifiées sont autorisées, mais ont une limite de débit inférieure à celle des extractions authentifiées. Pour vous authentifier à l'aide de votre AWS compte avant de le tirer, utilisez la commande suivante.
>
> ```
> aws ecr-public get-login-password --region us-east-1 | docker login --username
> AWS --password-stdin public.ecr.aws
> ```

Amazon ECR

L'image AWS for Fluent Bit est disponible sur Amazon ECR pour une haute disponibilité. Ces images sont disponibles dans la plupart des cas Régions AWS, y compris AWS GovCloud (US).

Linux

La dernière version stable de AWS l'URI d'image Fluent Bit peut être récupérée à l'aide de la commande suivante.

```
aws ssm get-parameters \
```

```
      --names /aws/service/aws-for-fluent-bit/stable \
      --region us-east-1
```

Toutes les versions de l'image AWS for Fluent Bit peuvent être répertoriées à l'aide de la commande suivante pour interroger le paramètre Systems Manager Parameter Store.

```
aws ssm get-parameters-by-path \
      --path /aws/service/aws-for-fluent-bit \
      --region us-east-1
```

L'image stable la plus récente AWS pour Fluent Bit peut être référencée dans un AWS CloudFormation modèle en faisant référence au nom du magasin de paramètres Systems Manager. Voici un exemple :

```
Parameters:
  FireLensImage:
    Description: Fluent Bit image for the FireLens Container
    Type: AWS::SSM::Parameter::Value<String>
    Default: /aws/service/aws-for-fluent-bit/stable
```

Windows

La dernière version stable de AWS l'URI d'image Fluent Bit peut être récupérée à l'aide de la commande suivante.

```
aws ssm get-parameters \
      --names /aws/service/aws-for-fluent-bit/windowsservercore-stable \
      --region us-east-1
```

Toutes les versions de l'image AWS for Fluent Bit peuvent être répertoriées à l'aide de la commande suivante pour interroger le paramètre Systems Manager Parameter Store.

```
aws ssm get-parameters-by-path \
      --path /aws/service/aws-for-fluent-bit/windowsservercore \
      --region us-east-1
```

La dernière image stable AWS pour Fluent Bit peut être référencée dans un AWS CloudFormation modèle en faisant référence au nom du magasin de paramètres Systems Manager. Voici un exemple :

```
Parameters:
  FireLensImage:
    Description: Fluent Bit image for the FireLens Container
    Type: AWS::SSM::Parameter::Value<String>
    Default: /aws/service/aws-for-fluent-bit/windowsservercore-stable
```

Exemple de définition de tâche Amazon ECS : acheminer les journaux vers FireLens

Pour utiliser le routage des journaux personnalisé avec FireLens, vous devez spécifier les éléments suivants dans votre définition de tâche :

- Conteneur de routeur journal contenant une configuration FireLens. Nous recommandons que le conteneur soit marqué comme `essential`.
- Un ou plusieurs conteneurs d'application contenant une configuration de journal spécifiant le pilote de journal `awsfirelens`.
- Un rôle IAM de tâche Amazon Resource Name (ARN) qui contient les autorisations nécessaires à la tâche pour acheminer les journaux.

Lorsque vous créez une nouvelle définition de tâche à l'aide de AWS Management Console, il existe une section FireLens d'intégration qui facilite l'ajout d'un conteneur de routeurs de journaux. Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.

Amazon ECS convertit la configuration du journal et génère la configuration de sortie Fluentd ou Fluent Bit. La configuration de sortie est montée dans le conteneur de routage des journaux à `/fluent-bit/etc/fluent-bit.conf` pour Fluent Bit et `/fluentd/etc/fluent.conf` pour Fluentd.

> **Important**
>
> FireLens écoute sur le port `24224`. Par conséquent, pour garantir que le routeur de FireLens journaux n'est pas accessible en dehors de la tâche, vous ne devez pas autoriser le trafic entrant sur le port `24224` du groupe de sécurité utilisé par votre tâche. Pour les tâches utilisant le mode réseau `awsvpc`, il s'agit du groupe de sécurité associé à la tâche. Pour les tâches qui utilisent le mode `host` réseau, il s'agit du groupe de sécurité associé à l' EC2 instance Amazon hébergeant la tâche. Pour les tâches utilisant le mode réseau `bridge`, ne créez pas de mappages de ports utilisant le port `24224`.

Par défaut, Amazon ECS ajoute des champs supplémentaires dans vos entrées de journal qui aident à identifier la source des journaux.

- `ecs_cluster` : nom du cluster dont la tâche fait partie.
- `ecs_task_arn` : Amazon Resource Name (ARN) de la tâche dont le conteneur fait partie.
- `ecs_task_definition` : nom et révision de la définition de tâche que la tâche utilise.
- `ec2_instance_id`— L'ID d' EC2 instance Amazon sur lequel le conteneur est hébergé. Ce champ n'est valide que pour les tâches utilisant le type de EC2 lancement.

Vous pouvez définir la valeur `enable-ecs-log-metadata` sur `false` si vous ne souhaitez pas obtenir les métadonnées.

L'exemple de définition de tâche suivant définit un conteneur de routeur de journaux qui utilise Fluent Bit pour acheminer ses CloudWatch journaux vers Logs. Il définit également un conteneur d'applications qui utilise une configuration de journal pour acheminer les journaux vers Amazon Data Firehose et définit la mémoire utilisée pour mettre en mémoire tampon les événements à 2 Mo.

> **Note**
>
> Pour plus d'exemples de définitions de tâches, consultez les FireLens exemples d'Amazon ECS sur GitHub.

```
{
  "family": "firelens-example-firehose",
  "taskRoleArn": "arn:aws:iam::123456789012:role/ecs_task_iam_role",
  "containerDefinitions": [
    {
            "name": "log_router",
            "image": "public.ecr.aws/aws-observability/aws-for-fluent-bit:stable",
            "cpu": 0,
            "memoryReservation": 51,
            "portMappings": [],
            "essential": true,
            "environment": [],
            "mountPoints": [],
            "volumesFrom": [],
            "user": "0",
            "logConfiguration": {
```

```
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/ecs-aws-firelens-sidecar-container",
                    "mode": "non-blocking",
                    "awslogs-create-group": "true",
                    "max-buffer-size": "25m",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "firelens"
                },
                "secretOptions": []
            },
            "systemControls": [],
            "firelensConfiguration": {
                "type": "fluentbit"
            }
        },
    {
      "essential": true,
      "image": "public.ecr.aws/docker/library/httpd:latest",
      "name": "app",
      "logConfiguration": {
        "logDriver": "awsfirelens",
        "options": {
          "Name": "firehose",
          "region": "us-west-2",
          "delivery_stream": "my-stream",
          "log-driver-buffer-limit": "2097152"
        }
      },
      "memoryReservation": 100
    }
  ]
}
```

Les paires clés/valeurs spécifiées en tant qu'options dans l'objet `logConfiguration` sont utilisées pour générer la configuration de sortie Fluentd ou Fluent Bit. Voici un exemple de code à partir d'une définition de sortie Fluent Bit.

```
[OUTPUT]
    Name   firehose
    Match  app-firelens*
    region us-west-2
```

```
    delivery_stream my-stream
```

> **Note**
>
> FireLens gère la `match` configuration. Vous ne spécifiez pas la `match` configuration dans votre définition de tâche.

Utiliser un fichier de configuration personnalisé

Vous pouvez spécifier un fichier de configuration personnalisé. Le format du fichier de configuration est le format natif du routeur de journal que vous utilisez. Pour plus d'informations, consultez Syntaxe du fichier de configuration Fluentd et configuration YAML.

Dans votre fichier de configuration personnalisé, pour les tâches utilisant le mode réseau `bridge` ou `awsvpc`, ne définissez pas d'entrée de transfert Fluentd ou Fluent Bit sur TCP, car FireLens l'ajoute à la configuration d'entrée.

Votre configuration FireLens doit contenir les options suivantes pour spécifier un fichier de configuration personnalisé :

`config-file-type`

Emplacement source du fichier de configuration personnalisé. Les options disponibles sont `s3` ou `file`.

> **Note**
>
> Les tâches hébergées sur AWS Fargate ne prennent en charge que le type `file` de fichier de configuration.

`config-file-value`

Source du fichier de configuration personnalisé. Si le type de fichier de configuration `s3` est utilisé, la valeur du fichier de configuration est l'ARN complet du compartiment Amazon S3 et du fichier. Si le type de fichier de configuration `file` est utilisé, la valeur du fichier de configuration est le chemin d'accès complet du fichier de configuration qui existe soit dans l'image du conteneur, soit sur un volume monté dans le conteneur.

> **Important**
>
> Lorsque vous utilisez un fichier de configuration personnalisé, vous devez spécifier un chemin différent de celui utilisé par FireLens. Amazon ECS réserve les chemins de fichier `/fluent-bit/etc/fluent-bit.conf` pour Fluent Bit et `/fluentd/etc/fluent.conf` pour Fluentd.

L'exemple suivant montre la syntaxe requise lors de la spécification d'une configuration personnalisée.

> **Important**
>
> Pour spécifier un fichier de configuration personnalisé hébergé dans Amazon S3, vérifiez que vous avez créé un rôle IAM d'exécution de tâche avec les autorisations appropriées.

Voici la syntaxe requise lors de la spécification d'une configuration personnalisée.

```
{
  "containerDefinitions": [
    {
      "essential": true,
      "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-
bit:stable",
      "name": "log_router",
      "firelensConfiguration": {
        "type": "fluentbit",
        "options": {
          "config-file-type": "s3 | file",
          "config-file-value": "arn:aws:s3:::amzn-s3-demo-bucket/fluent.conf
 | filepath"
        }
      }
    }
  ]
}
```

> **Note**
>
> Les tâches hébergées sur AWS Fargate ne prennent en charge que le type `file` de fichier de configuration.

Utilisation d'images autres que des AWS conteneurs dans Amazon ECS

Utilisez le registre privé pour stocker vos informations d'identification AWS Secrets Manager, puis référencez-les dans votre définition de tâche. Cela permet de référencer des images de conteneurs qui existent dans des registres privés en dehors de ceux AWS qui nécessitent une authentification dans vos définitions de tâches. Cette fonctionnalité est prise en charge par les tâches hébergées sur Fargate, les instances EC2 Amazon et les instances externes utilisant Amazon ECS Anywhere.

> **Important**
>
> Si votre définition de tâche fait référence à une image stockée dans Amazon ECR, cette rubrique ne s'applique pas. Pour plus d'informations, consultez Utilisation d'images Amazon ECR avec Amazon ECS dans le Guide de l'utilisateur Amazon Elastic Container Registry.

Pour les tâches hébergées sur EC2 des instances Amazon, cette fonctionnalité nécessite une version `1.19.0` ou une version ultérieure de l'agent de conteneur. Cependant, nous vous recommandons d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Pour les tâches hébergées sur Fargate, cette fonction nécessite une version `1.2.0` de la plateforme ou une version ultérieure. Pour plus d'informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.

Dans votre définition du conteneur, spécifiez l'objet `repositoryCredentials` avec les détails du secret que vous avez créé. Le secret référencé peut provenir d'un compte différent Région AWS ou différent de celui de la tâche qui l'utilise.

> **Note**
>
> Lorsque vous utilisez l'API ou le AWS SDK Amazon ECS, si le secret existe dans la même tâche Région AWS que la tâche que vous lancez, vous pouvez utiliser l'ARN complet ou le

nom du secret. AWS CLI Si le secret existe dans un autre compte, l'ARN complet du secret doit être spécifié. Lorsque vous utilisez le AWS Management Console, l'ARN complet du secret doit toujours être spécifié.

Voici un extrait de code d'une définition de tâche indiquant les paramètres requis :

Remplacez les paramètres suivants :

- *private-repo*avec le nom d'hôte du dépôt privé
- *private-image*avec le nom de l'image
- *arn:aws:secretsmanager:region:aws_account_id:secret:secret_name*avec le nom secret Amazon Resource Name (ARN)

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
        }
    }
]
```

> **Note**
>
> Une autre méthode d'activation de l'authentification de registre privé utilise des variables d'environnement d'agent de conteneur Amazon ECS pour s'authentifier auprès de registres privés. Cette méthode n'est prise en charge que pour les tâches hébergées sur EC2 des instances Amazon. Pour de plus amples informations, veuillez consulter Configuration des instances de conteneur Amazon ECS pour les images Docker privées .

Pour utiliser un registre privé

1. La définition de tâche doit avoir un rôle d'exécution de tâche. Cela permet à l'agent de conteneur d'extraire l'image de conteneur. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

L'authentification par registre privé permet à vos tâches Amazon ECS d'extraire des images de conteneurs à partir de registres privés extérieurs AWS (tels que Docker Hub, Quay.io ou votre propre registre privé) qui nécessitent des informations d'authentification. Cette fonctionnalité utilise Secrets Manager pour stocker en toute sécurité vos informations d'identification de registre, qui sont ensuite référencées dans votre définition de tâche à l'aide du `repositoryCredentials` paramètre.

Pour plus d'informations sur la configuration de l'authentification du registre privé, consultez la section Utilisation d'images non AWS conteneurisées dans Amazon ECS.

Pour donner accès aux secrets qui contiennent vos informations d'identification de registre privées, ajoutez les autorisations suivantes en tant que politique intégrée au rôle d'exécution des tâches. Pour plus d'informations, consultez Ajout et suppression de politiques IAM.

- `secretsmanager:GetSecretValue`—Nécessaire pour récupérer les informations d'identification du registre privé auprès de Secrets Manager.
- `kms:Decrypt` : obligatoire uniquement si votre secret utilise une clé KMS personnalisée et non la clé par défaut. L'Amazon Resource Name (ARN) de votre clé personnalisée doit être ajouté en tant que ressource.

Voici un exemple de stratégie en ligne qui ajoute les autorisations.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [

 "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:secret_name",
                "arn:aws:kms:<region>:<aws_account_id>:key/key_id"
            ]
        }
    ]
}
```

2. AWS Secrets Manager À utiliser pour créer un secret pour vos informations d'identification de registre privées. Pour plus d'informations sur la création d'un secret, voir Création d'un AWS Secrets Manager secret dans le Guide de AWS Secrets Manager l'utilisateur.

   Entrez les informations d'identification de votre registre privé au format suivant :

   ```
   {
     "username" : "privateRegistryUsername",
     "password" : "privateRegistryPassword"
   }
   ```

3. Enregistrer une définition de tâche. Pour de plus amples informations, veuillez consulter the section called “Création d'une définition de tâche à l'aide de la console”.

Redémarrer des conteneurs individuels dans des tâches Amazon ECS avec des politiques de redémarrage de conteneurs

Vous pouvez activer une politique de redémarrage pour chaque conteneur essentiel et non essentiel défini dans votre définition de tâche, afin de surmonter plus rapidement les défaillances transitoires et de maintenir la disponibilité des tâches. Lorsque vous activez une politique de redémarrage pour un conteneur, Amazon ECS peut redémarrer le conteneur s'il existe, sans avoir à remplacer la tâche.

Les politiques de redémarrage ne sont pas activées pour les conteneurs par défaut. Lorsque vous activez une politique de redémarrage pour un conteneur, vous pouvez spécifier des codes de sortie sur lesquels le conteneur ne sera pas redémarré. Il peut s'agir de codes de sortie indiquant le succès, tels que des codes de sortie`0`, qui ne nécessitent pas de redémarrage. Vous pouvez également spécifier la durée pendant laquelle un conteneur doit fonctionner correctement avant qu'un redémarrage puisse être tenté. Pour obtenir plus d'informations sur ces paramètres, consultez Politique de redémarrage. Pour un exemple de définition de tâche qui spécifie ces valeurs, consultezSpécification d'une politique de redémarrage de conteneur dans une définition de tâche Amazon ECS.

Vous pouvez utiliser le point de terminaison des métadonnées des tâches Amazon ECS ou CloudWatch Container Insights pour surveiller le nombre de redémarrages d'un conteneur. Pour plus d'informations sur le point de terminaison des métadonnées des tâches, consultez Point de terminaison des métadonnées des tâches Amazon ECS, version 4 etPoint de terminaison de métadonnées de tâches Amazon ECS version 4 pour les tâches sur Fargate. Pour plus d'informations

sur les métriques Container Insights pour Amazon ECS, consultez les métriques Amazon ECS Container Insights dans le guide de CloudWatch l'utilisateur Amazon.

Les politiques de redémarrage des conteneurs sont prises en charge par les tâches hébergées sur Fargate, les instances EC2 Amazon et les instances externes utilisant Amazon ECS Anywhere.

Considérations

Tenez compte des points suivants avant d'activer une politique de redémarrage pour votre conteneur :

- Les politiques de redémarrage ne sont pas prises en charge pour les conteneurs Windows sur Fargate.
- Pour les tâches hébergées sur EC2 des instances Amazon, cette fonctionnalité nécessite une version `1.86.0` ou une version ultérieure de l'agent de conteneur. Cependant, nous vous recommandons d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.
- Pour les tâches hébergées sur Fargate, cette fonction nécessite une version `1.4.0` de la plateforme ou une version ultérieure. Pour plus d'informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.
- Si vous utilisez le type de EC2 lancement avec le mode `bridge` réseau, la variable d'`FLUENT_HOST`environnement de votre conteneur d'applications peut devenir inexacte après le redémarrage du conteneur FireLens log router (le conteneur contenant l'`firelensConfiguration`objet dans sa définition de conteneur). Cela `FLUENT_HOST` est dû au fait qu'il s'agit d'une adresse IP dynamique qui peut changer après un redémarrage. La journalisation directe depuis le conteneur de l'application vers l'adresse `FLUENT_HOST` IP peut commencer à échouer après le changement d'adresse. Pour plus d'informations sur `FLUENT_HOST`, consultez Configuration des journaux Amazon ECS pour un débit élevé.
- L'agent Amazon ECS gère les politiques de redémarrage des conteneurs. Si, pour une raison inattendue, l'agent Amazon ECS échoue ou ne fonctionne plus, le conteneur ne sera pas redémarré.
- La période de tentative de redémarrage définie dans votre politique détermine la période (en secondes) pendant laquelle le conteneur doit fonctionner avant qu'Amazon ECS ne redémarre un conteneur.

Spécification d'une politique de redémarrage de conteneur dans une définition de tâche Amazon ECS

Pour spécifier une politique de redémarrage pour un conteneur dans une définition de tâche, dans la définition du conteneur, spécifiez l'`restartPolicy`objet. Pour plus d'informations sur l'`restartPolicy`objet, consultez[Politique de redémarrage](#).

Voici une définition de tâche utilisant le type de lancement de conteneurs Linux sur Fargate qui configure un serveur Web. La définition du conteneur inclut l'`restartPolicy`objet, `enabled` défini sur true pour activer une politique de redémarrage pour le conteneur. Le conteneur doit fonctionner pendant 180 secondes avant de pouvoir être redémarré et ne sera pas redémarré s'il sort avec le code de sortie`0`, ce qui indique un succès.

```
{
  "containerDefinitions": [
    {
      "command": [
        "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</title>
 <style>body {margin-top: 40px; background-color: #333;} </style> </head><body>
 <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
      ],
      "entryPoint": ["sh", "-c"],
      "essential": true,
      "image": "public.ecr.aws/docker/library/httpd:2.4",
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/fargate-task-definition",
          "awslogs-region": "us-east-1",
          "awslogs-stream-prefix": "ecs"
        }
      },
      "name": "sample-fargate-app",
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80,
          "protocol": "tcp"
        }
```

```
      ],
      "restartPolicy": {
        "enabled": true,
        "ignoredExitCodes": [0],
        "restartAttemptPeriod": 180
      }
    }
  ],
  "cpu": "256",
  "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
  "family": "fargate-task-definition",
  "memory": "512",
  "networkMode": "awsvpc",
  "runtimePlatform": {
    "operatingSystemFamily": "LINUX"
  },
  "requiresCompatibilities": ["FARGATE"]
}
```

Après avoir enregistré une définition de tâche avec l'`restartPolicy`objet dans une définition de conteneur, vous pouvez exécuter une tâche ou créer un service avec cette définition de tâche. Pour plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS et Création d'un déploiement de mises à jour continues Amazon ECS.

Transférer des données sensibles vers un conteneur Amazon ECS

Vous pouvez transmettre en toute sécurité des données sensibles, telles que des informations d'identification vers une base de données, dans votre conteneur.

Les secrets, tels que les clés d'API et les informations d'identification de base de données, sont fréquemment utilisés par les applications pour accéder à d'autres systèmes. Ils se composent souvent d'un nom d'utilisateur et d'un mot de passe, d'un certificat ou d'une clé d'API. L'accès à ces secrets doit être limité à des principaux IAM spécifiques qui utilisent IAM et injectés dans des conteneurs lors de l'exécution.

Les secrets peuvent être injectés facilement dans des conteneurs depuis AWS Secrets Manager Amazon EC2 Systems Manager Parameter Store. Ces secrets peuvent être référencés dans votre tâche de l'une des manières suivantes.

1. Ils sont référencés en tant que variables d'environnement qui utilisent le paramètre de définition du conteneur `secrets`.

2. Ils sont référencés comme `secretOptions` si votre plateforme de journalisation nécessite une authentification. Pour plus d'informations, veuillez consulter Options de configuration de la journalisation (langue française non garantie).
3. Ils sont référencés comme des secrets extraits par des images qui utilisent le paramètre de définition de conteneur `repositoryCredentials` si le registre d'où le conteneur est extrait nécessite une authentification. Utilisez cette méthode lors de l'extraction d'images depuis la Galerie publique Amazon ECR. Pour plus d'informations, veuillez consulter Authentification de registre privé pour les tâches (langue française non garantie).

Nous vous recommandons d'effectuer les opérations suivantes lorsque vous configurez la gestion des secrets.

Utiliser AWS Secrets Manager ou AWS Systems Manager Parameter Store pour stocker des documents secrets

Vous devez stocker en toute sécurité les clés d'API, les informations d'identification de base de données et les autres éléments secrets dans Secrets Manager ou sous forme de paramètre chiffré dans le magasin de paramètres de Systems Manager. Ces services sont similaires car il s'agit tous deux de magasins de valeurs clés gérés utilisés AWS KMS pour chiffrer des données sensibles. Cependant, Secrets Manager inclut également la possibilité de faire pivoter automatiquement les secrets, de générer des secrets aléatoires et de partager des secrets entre les comptes. Pour utiliser ces fonctionnalités, utilisez Secrets Manager. Dans le cas contraire, utilisez des paramètres chiffrés dans le magasin de paramètres de Systems Manager.

> **Important**
>
> Si votre secret change, vous devez forcer un nouveau déploiement ou lancer une nouvelle tâche pour récupérer la dernière valeur secrète. Pour plus d’informations, consultez les rubriques suivantes :
>
> - Tâches : arrêtez la tâche, puis démarrez-la. Pour plus d’informations, consultez Arrêt d'une tâche Amazon ECS et Exécution d'une application en tant que tâche Amazon ECS.
> - Service : mettez à jour le service et utilisez l'option Forcer un nouveau déploiement. Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.

Extraire les données d'un compartiment Amazon S3 chiffré

Vous devez stocker les secrets dans un compartiment Amazon S3 chiffré et utiliser des rôles de tâches pour restreindre l'accès à ces secrets. Cela empêche les valeurs des variables d'environnement de s'échapper par inadvertance dans les journaux et d'être révélées lors de l'exécution. `docker inspect` Dans ce cas, votre application doit être écrite pour lire le secret du compartiment Amazon S3. Pour obtenir des instructions, veuillez consulter Définition du comportement de chiffrement côté serveur par défaut pour les compartiments Amazon S3.

Montez le secret sur un volume à l'aide d'un conteneur sidecar

Comme le risque de fuite de données lié aux variables d'environnement est élevé, vous devez utiliser un conteneur annexe qui lit vos secrets AWS Secrets Manager et les écrit sur un volume partagé. Ce conteneur peut s'exécuter et sortir avant le conteneur d'applications en utilisant le système de commande de conteneurs Amazon ECS. Ainsi, le conteneur de l'application monte ensuite le volume dans lequel le secret a été écrit. À l'instar de la méthode du compartiment Amazon S3, votre application doit être écrite pour lire le secret du volume partagé. Le volume étant limité à la tâche, il est automatiquement supprimé après l'arrêt de la tâche. Pour un exemple, consultez le projet task-def.json.

Sur Amazon EC2, le volume sur lequel le secret est écrit peut être chiffré à l'aide d'une clé gérée par AWS KMS le client. Activé AWS Fargate, le stockage en volume est automatiquement chiffré à l'aide d'une clé gérée par le service.

Transmettre une variable d'environnement individuelle à un conteneur Amazon ECS

> **Important**
>
> Nous vous recommandons de stocker vos données sensibles soit dans des AWS Secrets Manager secrets, soit dans des paramètres AWS Systems Manager Parameter Store. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.
>
> Les variables d'environnement spécifiées dans la définition de tâche sont lisibles par tous les utilisateurs et rôles autorisés à effectuer l'action `DescribeTaskDefinition` pour la définition de tâche.

Vous pouvez transmettre des variables d'environnement à vos conteneurs des manières suivantes :

- Individuellement à l'aide du paramètre de définition de conteneur `environment`. Cela correspond à l'option `--env` de docker container run.
- En bloc, en utilisant le paramètre de définition de conteneur `environmentFiles` pour répertorier un ou plusieurs fichiers contenant les variables d'environnement. Le fichier doit être hébergé dans Amazon S3. Cela correspond à l'option `--env-file` de docker run.

Voici un extrait d'une définition de tâche montrant comment spécifier des variables d'environnement individuelles.

```
{
    "family": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            ...
            "environment": [
                {
                    "name": "variable",
                    "value": "value"
                }
            ],
            ...
        }
    ],
    ...
}
```

Transmettre des variables d'environnement à un conteneur Amazon ECS

> **Important**
>
> Nous vous recommandons de stocker vos données sensibles soit dans des AWS Secrets Manager secrets, soit dans des paramètres AWS Systems Manager Parameter Store. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.
>
> Les fichiers de variables d'environnement sont des objets dans Simple Storage Service (Amazon S3) et toutes les considérations de sécurité Simple Storage Service (Amazon S3) s'appliquent.

Vous ne pouvez pas utiliser le `environmentFiles` paramètre sur les conteneurs Windows ni sur les conteneurs Windows sur Fargate.

Vous pouvez créer un fichier de variables d'environnement et le stocker dans Amazon S3 pour transmettre des variables d'environnement à votre conteneur.

En spécifiant des variables d'environnement dans un fichier, vous pouvez injecter en bloc des variables d'environnement. Dans votre définition de conteneur, spécifiez l'objet `environmentFiles` avec une liste de compartiments Amazon S3 contenant vos fichiers de variables d'environnement.

Amazon ECS n'impose pas de limite de taille aux variables d'environnement, mais un fichier de variables d'environnement volumineux peut occuper l'espace disque. Chaque tâche qui utilise un fichier de variables d'environnement entraîne le téléchargement d'une copie du fichier sur le disque. Amazon ECS supprime le fichier dans le cadre du nettoyage des tâches.

Pour plus d'informations sur les variables d'environnement prises en charge, veuillez consulter Paramètres de définition de conteneur avancés - Environnement.

Tenez compte des éléments suivants lors de la spécification d'un fichier de variable d'environnement dans une définition de conteneur.

- Pour les tâches Amazon ECS sur Amazon EC2, vos instances de conteneur nécessitent que l'agent de conteneur soit une version `1.39.0` ou une version ultérieure pour utiliser cette fonctionnalité. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.
- Pour les tâches Amazon ECS sur AWS Fargate, vos tâches doivent utiliser la version de plate-forme ou une `1.4.0` version ultérieure (Linux) pour utiliser cette fonctionnalité. Pour de plus amples informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.

  Vérifiez que la variable est prise en charge par la plateforme du système d'exploitation. Pour plus d'informations, consultez the section called “Définitions de conteneur” et the section called “Autres paramètres de définition de tâche”.
- Le fichier doit utiliser l'extension de fichier `.env` et l'encodage UTF-8.
- Le rôle d'exécution des tâches est requis pour utiliser cette fonctionnalité avec les autorisations supplémentaires pour Amazon S3. Cela permet à l'agent de conteneur d'extraire le fichier de variable d'environnement à partir d'Amazon S3. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

- Il y a une limite de 10 fichiers par définition de tâche.
- Chaque ligne d'un fichier d'environnement doit contenir une variable d'environnement au format `VARIABLE=VALUE`. Des espaces ou des guillemets sont inclus dans les valeurs pour les fichiers Amazon ECS. Les lignes commençant par # sont traitées comme des commentaires et sont ignorées. Pour plus d'informations sur la syntaxe des fichiers de variables d'environnement, consultez la section Définir les variables d'environnement (-e, --env, --env-file) dans la documentation. Docker

  Voici la syntaxe appropriée.

  ```
  #This is a comment and will be ignored
  VARIABLE=VALUE
  ENVIRONMENT=PRODUCTION
  ```

- Si des variables d'environnement sont spécifiées à l'aide du paramètre `environment` dans une définition de conteneur, elles ont priorité sur les variables contenues dans un fichier d'environnement.
- Si plusieurs fichiers d'environnement contenant la même variable sont spécifiés, ils sont traités par ordre d'entrée. Cela signifie que la première valeur de la variable est utilisée et que les valeurs suivantes des variables dupliquées sont ignorées. Nous vous recommandons d'utiliser des noms de variables uniques.
- Si un fichier d'environnement est spécifié en tant que remplacement de conteneur, il est utilisé. De plus, tous les autres fichiers d'environnement spécifiés dans la définition du conteneur sont ignorés.
- Les règles suivantes s'appliquent au type de lancement Fargate :
  - Le fichier est géré de la même manière qu'un fichier d'environnement Docker natif.
  - Les définitions de conteneur qui font référence à des variables d'environnement vides et stockées dans Amazon S3 n'apparaissent pas dans le conteneur.
  - La gestion de l'échappement dans shell n'est pas prise en charge.
  - Le point d'entrée du conteneur interprète les valeurs `VARIABLE`.

exemple

Voici un extrait d'une définition de tâche indiquant comment spécifier un fichier de variable d'environnement.

```
{
    "family": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            ...
            "environmentFiles": [
                {
                    "value": "arn:aws:s3:::amzn-s3-demo-
bucket/envfile_object_name.env",
                    "type": "s3"
                }
            ],
            ...
        }
    ],
    ...
}
```

Transmettez les secrets de Secrets Manager par programmation dans Amazon ECS

Au lieu de coder en dur les informations sensibles en texte brut dans votre application, vous pouvez utiliser Secrets Manager pour stocker les données sensibles.

Nous recommandons cette méthode pour récupérer les données sensibles, car si le secret de Secrets Manager est ultérieurement mis à jour, l'application récupère automatiquement la dernière version de celui-ci.

Créez un secret dans Secrets Manager. Après avoir créé un secret dans Secrets Manager, mettez à jour le code de votre application pour récupérer le secret.

Prenez en compte les points suivants avant de sécuriser des données sensibles dans Secrets Manager.

- Seuls les secrets qui stockent des données texte, qui sont des secrets créés avec le `SecretString` paramètre de l'CreateSecretAPI, sont pris en charge. Les secrets qui stockent des données binaires, qui sont des secrets créés avec le `SecretBinary` paramètre de l'CreateSecretAPI, ne sont pas pris en charge.
- Utilisez les points de terminaison VPC de l'interface pour améliorer les contrôles de sécurité. Vous devez créer les points de terminaison d'un VPC pour Secrets Manager. Pour plus d'informations

sur le point de terminaison d'un VPC, veuillez consulter Créer des points de terminaison d'un VPC dans le Guide de l'utilisateur AWS Secrets Manager .

- Le VPC que votre tâche utilise doit utiliser la résolution DNS.
- Votre définition de tâche doit utiliser un rôle de tâche doté des autorisations supplémentaires pour Secrets Manager. Pour de plus amples informations, veuillez consulter Rôle IAM de la tâche Amazon ECS.

Créer le secret Secrets Manager

Vous pouvez utiliser la console Secrets Manager afin de créer un secret pour vos données sensibles. Pour plus d'informations sur la création de secrets, consultez la rubrique Création d'un secret AWS Secrets Manager dans le Guide de l'utilisateur AWS Secrets Manager .

Mettez à jour votre application pour récupérer le secret Secrets Manager par programme

Vous pouvez récupérer des secrets en appelant le Secrets Manager APIs directement depuis votre application. Pour plus d'informations, consultez la section Récupérer des secrets AWS Secrets Manager dans le guide de AWS Secrets Manager l'utilisateur.

Pour récupérer les données sensibles stockées dans le AWS Secrets Manager, consultez les exemples de code à AWS Secrets Manager utiliser AWS SDKs dans la bibliothèque de codes d'exemples de code du AWS SDK.

Transmettez les secrets du magasin de paramètres de Systems Manager par programmation dans Amazon ECS

Systems Manager Parameter Store offre un stockage et une gestion sécurisés des secrets. Vous pouvez stocker des données telles que les mots de passe, les chaînes de base de données, les EC2 instances IDs et les AMI IDs, ainsi que les codes de licence sous forme de valeurs de paramètres, au lieu de coder ces informations en dur dans votre application. Vous pouvez stocker ces valeurs sous forme de texte brut ou de données chiffrées.

Nous recommandons cette méthode pour récupérer les données sensibles car si le paramètre Systems Manager Parameter Store est ultérieurement mis à jour, l'application récupère automatiquement la dernière version.

Prenez en compte les points suivants avant de sécuriser des données sensibles dans Systems Manager Parameter Store.

- Seuls les secrets qui stockent des données texte sont pris en charge. Les secrets qui stockent des données binaires ne sont pas pris en charge.
- Utilisez les points de terminaison d'un VPC de l'interface pour améliorer les contrôles de sécurité.
- Le VPC que votre tâche utilise doit utiliser la résolution DNS.
- Pour les tâches qui utilisent le type de EC2 lancement, vous devez utiliser la variable de configuration de l'agent Amazon ECS `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` pour utiliser cette fonctionnalité. Vous pouvez l'ajouter au fichier `/etc/ecs/ecs.config` lors de la création de l'instance de conteneur, ou vous pouvez l'ajouter à une instance existante, puis redémarrer l'agent ECS. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.
- Votre définition de tâche doit utiliser un rôle de tâche doté des autorisations supplémentaires pour le magasin de paramètres Systems Manager. Pour de plus amples informations, veuillez consulter Rôle IAM de la tâche Amazon ECS.

Créer le paramètre

Vous pouvez utiliser la console System Manager pour créer un paramètre Systems Manager Parameter Store pour vos données sensibles. Pour plus d'informations, consultez Créer un paramètre Systems Manager (console) ou Créer un paramètre Systems Manager (AWS CLI) dans le Guide de l'utilisateur AWS Systems Manager .

Mettez à jour votre application pour récupérer par programme les secrets Systems Manager Parameter Store.

Pour récupérer les données sensibles stockées dans le paramètre Systems Manager Parameter Store, consultez les exemples de code à utiliser par Systems Manager AWS SDKs dans la bibliothèque de codes d'exemples de code du AWS SDK.

Transmettez les secrets de Secrets Manager via les variables d'environnement Amazon ECS

Lorsque vous injectez un secret en tant que variable d'environnement, vous pouvez spécifier le contenu complet d'un secret, une clé JSON spécifique dans un secret. Cela vous aide à contrôler les données sensibles exposées à votre conteneur. Pour plus d'informations sur le contrôle des versions secrètes, voir Que contient un secret de Secrets Manager ? dans le guide de AWS Secrets Manager l'utilisateur.

Les points suivants doivent être pris en compte lors de l'utilisation d'une variable d'environnement pour injecter un secret Secrets Manager dans un conteneur.

- Les données sensibles sont injectées dans votre conteneur lors du démarrage initial du conteneur. Si le secret est ensuite mis à jour ou fait l'objet d'une rotation, le conteneur ne reçoit pas la valeur mise à jour automatiquement. Vous devez lancer une nouvelle tâche ou, si votre tâche fait partie d'un service, vous pouvez mettre à jour le service et utiliser l'option Force new deployment (Forcer un nouveau déploiement) pour forcer le service à lancer une nouvelle tâche.
- Les applications qui s'exécutent sur le conteneur, les journaux du conteneur et les outils de débogage ont accès aux variables d'environnement.
- Pour les tâches Amazon ECS sur AWS Fargate, tenez compte des points suivants :
  - Pour injecter le contenu complet d'un secret en tant que variable d'environnement ou dans une configuration de journal, vous devez utiliser la version `1.3.0` ou ultérieure de la plateforme. Pour plus d'informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.
  - Pour injecter une clé JSON spécifique ou une version d'un secret en tant que variable d'environnement ou dans une configuration de journal, vous devez utiliser la version `1.4.0` de la plateforme ou une version ultérieure (Linux) ou `1.0.0` (Windows). Pour plus d'informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.
- Pour les tâches Amazon ECS sur EC2, les points suivants doivent être pris en compte :
  - Pour injecter un secret à l'aide d'une clé JSON spécifique ou d'une version d'un secret, votre instance de conteneur doit avoir la version `1.37.0` ou ultérieure de l'agent de conteneur. Cependant, nous vous recommandons d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

    Pour injecter le contenu complet d'un secret en tant que variable d'environnement ou pour injecter un secret dans une configuration de journal, votre instance de conteneur doit avoir la version `1.22.0` ou ultérieure de l'agent de conteneur.
- Utilisez les points de terminaison VPC de l'interface pour améliorer les contrôles de sécurité et connectez-vous à Secrets Manager via un sous-réseau privé. Vous devez créer les points de terminaison d'un VPC pour Secrets Manager. Pour plus d'informations sur le point de terminaison d'un VPC, veuillez consulter Créer des points de terminaison d'un VPC dans le Guide de l'utilisateur AWS Secrets Manager . Pour plus d'informations sur l'utilisation de Secrets Manager et d'Amazon VPC, consultez Comment se connecter au service Secrets Manager au sein d'un Amazon VPC.

- Pour les tâches Windows configurées pour utiliser le pilote de journalisation `awslogs`, vous devez également définir la variable d'environnement `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE` sur votre instance de conteneur. Utilisez la syntaxe suivante :

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster <cluster name> -EnableTaskIAMRole -LoggingDrivers
 '["json-file","awslogs"]'
</powershell>
```

- Votre définition de tâche doit utiliser un rôle d'exécution de tâche doté des autorisations supplémentaires pour Secrets Manager. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

Créez le secret AWS Secrets Manager .

Vous pouvez utiliser la console Secrets Manager afin de créer un secret pour vos données sensibles. Pour plus d'informations, voir Création d'un AWS Secrets Manager secret dans le guide de AWS Secrets Manager l'utilisateur.

Ajoutez la variable d'environnement à la définition du conteneur

Dans votre définition de conteneur, vous pouvez spécifier les éléments suivants :

- Objet `secrets` contenant le nom de la variable d'environnement à définir dans le conteneur
- Amazon Resource Name (ARN) du secret Secrets Manager
- Paramètres supplémentaires contenant les données sensibles à présenter au conteneur

L'exemple suivant montre la syntaxe complète qui doit être spécifiée pour le secret Secrets Manager.

```
arn:aws:secretsmanager:region:aws_account_id:secret:secret-name:json-key:version-
stage:version-id
```

La section suivante décrit les paramètres supplémentaires. Ces paramètres sont facultatifs, mais si vous ne les utilisez pas, vous devez inclure les deux-points `:` pour utiliser les valeurs par défaut. Des exemples sont donnés ci-dessous pour plus de contexte.

`json-key`

Spécifie le nom de la clé dans une paire clé-valeur avec la valeur que vous souhaitez définir comme valeur de variable d'environnement. Seules les valeurs au format JSON sont prises en charge. Si vous ne spécifiez pas de clé JSON, le contenu complet du secret est utilisé.

`version-stage`

Spécifie l'étiquette intermédiaire de la version d'un secret que vous souhaitez utiliser. Si une étiquette intermédiaire de version est spécifiée, vous ne pouvez pas spécifier d'ID de version. Si aucune étape de version n'est spécifiée, le comportement par défaut consiste à récupérer le secret avec l'étiquette `AWSCURRENT` intermédiaire.

Les étiquettes intermédiaires sont utilisées pour suivre les différentes versions d'un secret lorsqu'elles sont mises à jour ou font l'objet d'une rotation. Chaque version d'un secret a une ou plusieurs étiquettes intermédiaires et un ID.

`version-id`

Spécifie l'identifiant unique de la version du secret que vous souhaitez utiliser. Si un ID de version est spécifié, vous ne pouvez pas spécifier d'étiquette intermédiaire de version. Si aucun ID de version n'est spécifié, le comportement par défaut consiste à récupérer le secret avec l'étiquette `AWSCURRENT` intermédiaire.

IDs Les versions sont utilisées pour suivre les différentes versions d'un secret lors de leur mise à jour ou de leur rotation. Chaque version d'un secret a un ID. Pour plus d'informations, veuillez consulter la rubrique Concepts et termes clés pour AWS Secrets Manager dans le Guide de l'utilisateur AWS Secrets Manager .

Exemples de définitions de conteneur

Les exemples suivants montrent comment vous pouvez référencer des secrets Secrets Manager dans vos définitions de conteneur.

Example référencement d'un secret complet

Voici un extrait d'une définition de tâche montrant le format lorsque vous référencez le texte complet d'un secret Secrets Manager.

```
{
```

```
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
    }]
  }]
}
```

Pour accéder à la valeur de ce secret depuis le conteneur, vous devez appeler `$environment_variable_name`.

Example référencer tous les secrets

Ce qui suit est un extrait d'une définition de tâche indiquant le format lors de la référence au texte intégral de plusieurs secrets de Secrets Manager.

```
{
  "containerDefinitions": [{
     "secrets": [
      {
        "name": "environment_variable_name1",
         "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
      },
      {
        "name": "environment_variable_name2",
         "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
abcdef"
      },
      {
        "name": "environment_variable_name3",
        "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
ABCDEF"
      }
    ]
  }]
}
```

Pour accéder à la valeur de ce secret depuis le conteneur, vous devez appeler le `$environment_variable_name1`$environment_variable_name2, et$environment_variable_name3.

Example référencement d'une clé spécifique dans un secret

Voici un exemple de sortie d'une get-secret-valuecommande qui affiche le contenu d'un secret ainsi que le libellé de la version intermédiaire et l'ID de version qui lui sont associés.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "VersionId": "871d9eca-18aa-46a9-8785-981ddEXAMPLE",
    "SecretString": "{\"username1\":\"password1\",\"username2\":\"password2\",
\"username3\":\"password3\"}",
    "VersionStages": [
        "AWSCURRENT"
    ],
    "CreatedDate": 1581968848.921
}
```

Référence d'une clé spécifique de la sortie précédente dans une définition de conteneur en spécifiant le nom de la clé à la fin de l'ARN.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::"
    }]
  }]
}
```

Example référencement d'une version secrète spécifique

Voici un exemple de sortie d'une commande describe-secret qui affiche le contenu non chiffré d'un secret ainsi que les métadonnées de toutes les versions du secret.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "Description": "Example of a secret containing application authorization data.",
    "RotationEnabled": false,
    "LastChangedDate": 1581968848.926,
    "LastAccessedDate": 1581897600.0,
```

```
    "Tags": [],
    "VersionIdsToStages": {
        "871d9eca-18aa-46a9-8785-981ddEXAMPLE": [
            "AWSCURRENT"
        ],
        "9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE": [
            "AWSPREVIOUS"
        ]
    }
}
```

Référence d'une étiquette intermédiaire de version spécifique à partir de la sortie précédente dans une définition de conteneur en spécifiant le nom de la clé à la fin de l'ARN.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf::AWSPREVIOUS:"
    }]
  }]
}
```

Référence un ID de version spécifique de la sortie précédente dans une définition de conteneur en spécifiant le nom de clé à la fin de l'ARN.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:::9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE"
    }]
  }]
}
```

Example référencement d'une clé spécifique et d'une étiquette intermédiaire de version d'un secret

Ce qui suit montre comment référencer à la fois une clé spécifique dans une étiquette secrète et une étiquette de mise en scène de version spécifique.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1:AWSPREVIOUS:"
    }]
  }]
}
```

Pour spécifier une clé et un ID de version spécifiques, utilisez la syntaxe suivante.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE"
    }]
  }]
}
```

Pour plus d'informations sur la création d'une définition de tâche avec le secret spécifié dans une variable d'environnement, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Transmettre les paramètres de Systems Manager via les variables d'environnement Amazon ECS

Amazon ECS vous permet d'injecter des données sensibles dans vos conteneurs en les stockant dans les paramètres du AWS Systems Manager Parameter Store, puis en les référençant dans la définition de votre conteneur.

Tenez compte des points suivants lorsque vous utilisez une variable d'environnement pour injecter un secret Systems Manager dans un conteneur.

- Les données sensibles sont injectées dans votre conteneur lors du démarrage initial du conteneur. Si le secret est ensuite mis à jour ou fait l'objet d'une rotation, le conteneur ne reçoit pas la valeur mise à jour automatiquement. Vous devez lancer une nouvelle tâche ou, si votre tâche fait partie d'un service, vous pouvez mettre à jour le service et utiliser l'option Force new deployment (Forcer un nouveau déploiement) pour forcer le service à lancer une nouvelle tâche.

- Pour les tâches Amazon ECS sur AWS Fargate, les points suivants doivent être pris en compte :
  - Pour injecter le contenu complet d'un secret en tant que variable d'environnement ou dans une configuration de journal, vous devez utiliser la version `1.3.0` ou ultérieure de la plateforme. Pour plus d'informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.
  - Pour injecter une clé JSON spécifique ou une version d'un secret en tant que variable d'environnement ou dans une configuration de journal, vous devez utiliser la version `1.4.0` de la plateforme ou une version ultérieure (Linux) ou `1.0.0` (Windows). Pour plus d'informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.
- Pour les tâches Amazon ECS sur EC2, les points suivants doivent être pris en compte :
  - Pour injecter un secret à l'aide d'une clé JSON spécifique ou d'une version d'un secret, votre instance de conteneur doit avoir la version `1.37.0` ou ultérieure de l'agent de conteneur. Cependant, nous vous recommandons d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

    Pour injecter le contenu complet d'un secret en tant que variable d'environnement ou pour injecter un secret dans une configuration de journal, votre instance de conteneur doit avoir la version `1.22.0` ou ultérieure de l'agent de conteneur.
- Utilisez les points de terminaison VPC de l'interface pour améliorer les contrôles de sécurité. Vous devez créer les points de terminaison VPC de l'interface pour Systems Manager. Pour plus d'informations sur le point de terminaison VPC, consultez la section Améliorer la sécurité des EC2 instances en utilisant les points de terminaison VPC pour Systems Manager dans le guide de l'utilisateur.AWS Systems Manager
- Votre définition de tâche doit utiliser un rôle d'exécution de tâche doté des autorisations supplémentaires pour le magasin de paramètres Systems Manager. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.
- Pour les tâches Windows configurées pour utiliser le pilote de journalisation `awslogs`, vous devez également définir la variable d'environnement `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE` sur votre instance de conteneur. Utilisez la syntaxe suivante :

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster <cluster name> -EnableTaskIAMRole -LoggingDrivers
 '["json-file","awslogs"]'
```

```
</powershell>
```

Création du paramètre Systems Manager

Vous pouvez utiliser la console System Manager pour créer un paramètre Systems Manager Parameter Store pour vos données sensibles. Pour plus d'informations, consultez Créer un paramètre Systems Manager (console) ou Créer un paramètre Systems Manager (AWS CLI) dans le Guide de l'utilisateur AWS Systems Manager .

Ajoutez la variable d'environnement à la définition du conteneur

Dans la définition de votre conteneur dans la définition de la tâche, spécifiez `secrets` le nom de la variable d'environnement à définir dans le conteneur et l'ARN complet du paramètre Systems Manager Parameter Store contenant les données sensibles à présenter au conteneur. Pour de plus amples informations, veuillez consulter `secrets`.

Voici un extrait d'une définition de tâche montrant le format à utiliser lorsque vous référencez un paramètre Systems Manager Parameter Store. Si le paramètre Systems Manager Parameter Store existe dans la même région que la tâche que vous lancez, vous pouvez utiliser le nom ou l'ARN complet du paramètre. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }]
  }]
}
```

Pour plus d'informations sur la création d'une définition de tâche avec le secret spécifié dans une variable d'environnement, consultezCréation d'une définition de tâche Amazon ECS à l'aide de la console.

Mettez à jour votre application pour récupérer par programme les secrets Systems Manager Parameter Store.

Pour récupérer les données sensibles stockées dans le paramètre Systems Manager Parameter Store, consultez les exemples de code à utiliser par Systems Manager AWS SDKs dans la bibliothèque de codes d'exemples de code du AWS SDK.

Transmettez les secrets pour la configuration de journalisation Amazon ECS

Vous pouvez utiliser le `secretOptions` paramètre in `logConfiguration` pour transmettre les données sensibles utilisées pour la journalisation.

Vous pouvez enregistrer le secret dans Secrets Manager ou Systems Manager.

Utiliser le Gestionnaire de Secrets

Dans votre définition de conteneur, lorsque vous spécifiez une `logConfiguration`, vous pouvez spécifier `secretOptions` avec le nom de l'option du pilote de journal à définir dans le conteneur et l'ARN complet du secret Secrets Manager contenant les données sensibles à présenter au conteneur. Pour plus d'informations sur la création de secrets, consultez la section Créer un AWS Secrets Manager.

Voici un extrait d'une définition de tâche montrant le format lorsque vous référencez un secret Secrets Manager.

```
{
  "containerDefinitions": [{
    "logConfiguration": [{
      "logDriver": "splunk",
      "options": {
        "splunk-url": "https://your_splunk_instance:8088"
      },
      "secretOptions": [{
        "name": "splunk-token",
        "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
      }]
    }]
  }]
}
```

Ajoutez la variable d'environnement à la définition du conteneur

Dans votre définition de conteneur, spécifiez `secrets` avec le nom de la variable d'environnement à définir dans le conteneur et l'ARN complet du paramètre Systems Manager Parameter Store contenant les données sensibles à présenter au conteneur. Pour de plus amples informations, veuillez consulter `secrets`.

Voici un extrait d'une définition de tâche montrant le format à utiliser lorsque vous référencez un paramètre Systems Manager Parameter Store. Si le paramètre Systems Manager Parameter Store existe dans la même région que la tâche que vous lancez, vous pouvez utiliser le nom ou l'ARN complet du paramètre. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }]
  }]
}
```

Pour plus d'informations sur la création d'une définition de tâche avec le secret spécifié dans une variable d'environnement, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Utiliser Systems Manager

Vous pouvez injecter des données sensibles dans une configuration de journal. Dans votre définition de conteneur, lorsque vous spécifiez `logConfiguration`, vous pouvez spécifier `secretOptions` avec le nom de l'option du pilote de journal à définir dans le conteneur et l'ARN complet du paramètre Systems Manager Parameter Store contenant les données sensibles à présenter au conteneur.

> **Important**
>
> Si le paramètre Systems Manager Parameter Store existe dans la même région que la tâche que vous lancez, vous pouvez utiliser le nom ou l'ARN complet du paramètre. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

Voici un extrait d'une définition de tâche montrant le format à utiliser lorsque vous référencez un paramètre Systems Manager Parameter Store.

```
{
  "containerDefinitions": [{
    "logConfiguration": [{
      "logDriver": "fluentd",
      "options": {
```

```
        "tag": "fluentd demo"
      },
      "secretOptions": [{
        "name": "fluentd-address",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter:/parameter_name"
      }]
    }]
  }]
}
```

Spécification de données sensibles à l'aide des secrets Secrets Manager dans Amazon ECS

Amazon ECS vous permet d'injecter des données sensibles dans vos conteneurs en les stockant en AWS Secrets Manager secret, puis en les référençant dans la définition de votre conteneur. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.

Apprenez à créer un secret Secrets Manager, à le référencer dans une définition de tâche Amazon ECS, puis à vérifier qu'il fonctionne en interrogeant la variable d'environnement dans un conteneur affichant le contenu du secret.

Prérequis

Le didacticiel suppose de remplir les prérequis suivants :

- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur dispose des autorisations IAM requises pour créer les ressources Secrets Manager et Amazon ECS.

Étape 1 : Créer un secret Secrets Manager

Vous pouvez utiliser la console Secrets Manager afin de créer un secret pour vos données sensibles. Dans ce didacticiel, nous allons créer un secret de base pour stocker un nom d'utilisateur et un mot de passe à référencer ultérieurement dans un conteneur. Pour plus d'informations, voir Création d'un AWS Secrets Manager secret dans le guide de AWS Secrets Manager l'utilisateur.

Les key/value paires à stocker dans ce secret sont la valeur de la variable d'environnement dans votre conteneur à la fin du didacticiel.

Enregistrez l'ARN secret pour le référencer dans votre politique IAM d'exécution de tâche et dans la définition de tâche lors des étapes ultérieures.

Étape 2 : ajouter les autorisations secrètes au rôle d'exécution des tâches

Pour qu'Amazon ECS puisse récupérer les données sensibles de votre secret Secrets Manager, vous devez disposer des autorisations relatives aux secrets pour le rôle d'exécution de la tâche. Pour de plus amples informations, veuillez consulter Permissions de Secrets Manager ou de Systems Manager.

Étape 3 : Créer une définition de tâche

Vous pouvez utiliser la console Amazon ECS pour créer une définition de tâche qui fait référence à un secret Secrets Manager.

Pour créer une définition de tâche qui spécifie un secret

Utilisez la console IAM pour mettre à jour votre rôle d'exécution de tâche avec les autorisations requises.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
4. Dans la zone de l'éditeur JSON, entrez le texte JSON de définition de tâche suivant, en vous assurant de spécifier l'ARN complet du secret Secrets Manager que vous avez créé à l'étape 1 et le rôle d'exécution de tâche que vous avez mis à jour à l'étape 2. Choisissez Enregistrer.
5.

```
{
    "executionRoleArn": "arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole",
    "containerDefinitions": [
        {
            "entryPoint": [
                "sh",
                "-c"
            ],
            "portMappings": [
                {
                    "hostPort": 80,
                    "protocol": "tcp",
```

```
                    "containerPort": 80
                }
            ],
            "command": [
                "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
            ],
            "cpu": 10,
            "secrets": [
                {
                    "valueFrom":
 "arn:aws:secretsmanager:region:aws_account_id:secret:username_value",
                    "name": "username_value"
                }
            ],
            "memory": 300,
            "image": "public.ecr.aws/docker/library/httpd:2.4",
            "essential": true,
            "name": "ecs-secrets-container"
        }
    ],
    "family": "ecs-secrets-tutorial"
}
```

6. Choisissez Créer.

Étape 4 : Création d'un cluster

Vous pouvez utiliser la console Amazon ECS pour créer un cluster contenant une instance de conteneur pour exécuter la tâche. Si vous avez un cluster existant avec au moins une instance de conteneur enregistrée avec les ressources disponibles pour exécuter une instance de la définition de tâche créée pour ce didacticiel, vous pouvez passer à l'étape suivante.

Pour ce didacticiel, nous allons créer un cluster avec une instance de conteneur `t2.micro` à l'aide de l'AMI Amazon Linux 2 optimisée pour Amazon ECS.

Pour plus d'informations sur la création d'un cluster pour le type de EC2 lancement, consultezthe section called “Création d'un cluster pour le type de EC2 lancement Amazon”.

Étape 5 : Exécuter une tâche

Vous pouvez utiliser la console Amazon ECS pour exécuter une tâche avec la définition de tâche que vous avez créée. Pour ce didacticiel, nous allons exécuter une tâche en utilisant le type de EC2 lancement, en utilisant le cluster que nous avons créé à l'étape précédente.

Pour plus d'informations sur l'exécution d'une tâche, consultez the section called “Exécution d'une application en tant que tâche”.

Étape 6 : Vérification

Vous pouvez vérifier que toutes les étapes ont été effectuées avec succès et que la variable d'environnement a été créée correctement dans votre conteneur en suivant les étapes ci-dessous.

Vérifier que la variable d'environnement a été créée

1. Trouvez l'adresse IP publique ou DNS pour votre instance de conteneur.
   a. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
   b. Dans le volet de navigation, choisissez Clusters, puis choisissez le cluster que vous avez créé.
   c. Choisissez Infrastructure, puis choisissez l'instance de conteneur.
   d. Enregistrez l'adresse Public IP (IP publique) ou Public DNS (DNS public) de votre instance.
2. Si vous utilisez un ordinateur MacOS ou Linux, connectez-vous à votre instance avec la commande suivante, en indiquant le chemin d'accès de votre clé privée et l'adresse publique de votre instance :

```
$ ssh -i /path/to/my-key-pair.pem ec2-user@ec2-198-51-100-1.compute-1.amazonaws.com
```

   Pour plus d'informations sur l'utilisation d'un ordinateur Windows, consultez Connect to your Linux instance using PuTTY dans le guide de EC2 l'utilisateur Amazon.

   > **Important**
   > Pour plus d'informations sur les problèmes rencontrés lors de la connexion à votre instance, consultez la section Résolution des problèmes liés à la connexion à votre instance dans le guide de EC2 l'utilisateur Amazon.

3. Répertoriez les conteneurs en cours d'exécution sur l'instance. Notez l'ID de conteneur pour le conteneur `ecs-secrets-tutorial`.

```
docker ps
```

4. Connectez-vous au conteneur `ecs-secrets-tutorial` à l'aide de l'ID conteneur à partir du résultat de l'étape précédente.

```
docker exec -it container_ID /bin/bash
```

5. Utilisez la commande `echo` pour imprimer la valeur de la variable d'environnement.

```
echo $username_value
```

Si le didacticiel s'est correctement déroulé, vous devriez voir le résultat suivant :

```
password_value
```

> **Note**
>
> Sinon, vous pouvez répertorier toutes les variables d'environnement dans votre conteneur à l'aide de la commande `env` (ou `printenv`).

Étape 7 : nettoyer

Une fois que vous avez terminé ce didacticiel, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources inutilisées.

Nettoyer les ressources.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Choisissez Delete Cluster (Supprimer le cluster).
5. Dans la zone de confirmation, saisissez Supprimer ***cluster name***, puis choisissez Supprimer.
6. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.

7. Dans le panneau de navigation, choisissez Roles (Rôles).
8. Recherchez `ecsTaskExecutionRole` dans la liste des rôles et sélectionnez-le.
9. Choisissez Autorisations, puis le X à côté de ECSSecretsTutoriel. Sélectionnez Remove (Supprimer).
10. Ouvrez la console Secrets Manager à l'adresse https://console.aws.amazon.com/secretsmanager/.
11. Sélectionnez le secret username_value que vous avez créé, puis choisissez Actions, Delete secret (Supprimer le secret).

Paramètres de définition des tâches Amazon ECS pour le type de lancement Fargate

Les définitions de tâches sont divisées en plusieurs parties : la famille de tâches, le rôle de tâche AWS Identity and Access Management (IAM), le mode réseau, les définitions de conteneurs, les volumes et les types de lancement. Les définitions de famille et de conteneur sont requises dans une définition de tâche. En revanche, le rôle de la tâche, le mode réseau, les volumes et le type de lancement sont facultatifs.

Vous pouvez utiliser ces paramètres dans un fichier JSON pour configurer votre définition de tâche.

Vous trouverez ci-dessous des descriptions plus détaillées de chaque paramètre de définition de tâche pour le type de lancement Fargate.

Famille

`family`

Type : String

Obligatoire : oui

Lorsque vous enregistrez une définition de tâche, vous lui attribuez une famille. Cela équivaut, pour plusieurs versions de définition de tâche, à lui attribuer un nom spécifié avec un numéro de révision. La première définition de tâche enregistrée dans une famille donnée reçoit le numéro de révision 1. Toute définition de tâche enregistrée après celle-ci reçoit un numéro de révision ultérieur dans l'ordre séquentiel.

Types de lancement

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier un type de lancement par rapport auquel Amazon ECS doit valider la définition de tâche. Une exception client est renvoyée si la définition de tâche n'est pas conforme aux compatibilités spécifiées. Pour de plus amples informations, veuillez consulter Types de lancement Amazon ECS.

Le paramètre suivant est autorisé dans une définition de tâche.

`requiresCompatibilities`

Type : tableau de chaînes

Obligatoire : non

Valeurs valides : `FARGATE`

Les types de lancement pour lesquels la définition de tâche doit être validée. Cela permet de lancer une vérification afin de s'assurer que tous les paramètres utilisés dans la définition de la tâche correspondent aux exigences du type de lancement.

Rôle de tâche

`taskRoleArn`

Type : chaîne

Obligatoire : non

Lorsque vous enregistrez une définition de tâche, vous pouvez attribuer un rôle de tâche à un rôle IAM qui permet aux conteneurs concernés par l'autorisation de tâche d'appeler AWS APIs les conteneurs spécifiés dans les politiques associées en votre nom. Pour de plus amples informations, veuillez consulter Rôle IAM de la tâche Amazon ECS.

Rôle d'exécution de tâche

`executionRoleArn`

Type : String

Obligatoire : Conditionnelle

Nom de ressource Amazon (ARN) du rôle d'exécution des tâches qui autorise l'agent de conteneur Amazon ECS à effectuer des appels d' AWS API en votre nom.

> **Note**
>
> Le rôle IAM d'exécution de la tâche est requis en fonction des besoins de votre tâche. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

Mode réseau

`networkMode`

Type : String

Obligatoire : oui

Mode réseau Docker à utiliser pour les conteneurs de la tâche. Pour les tâches Amazon ECS hébergées sur Fargate, `awsvpc` le mode réseau est requis.

Plateforme d'exécution

`operatingSystemFamily`

Type : String

Obligatoire : Conditionnelle

Par défaut : LINUX

Ce paramètre est requis pour les tâches Amazon ECS hébergées sur Fargate.

Lorsque vous enregistrez une définition de tâche, vous spécifiez la famille du système d'exploitation.

Les valeurs valides sont `LINUXWINDOWS_SERVER_2025_FULL`,`WINDOWS_SERVER_2025_CORE`, `WINDOWS_SERVER_2022_FULLWINDOWS_SERVER_2022_CORE`,`WINDOWS_SERVER_2019_FULL`, et`WINDOWS_SERVER_2019_CORE`.

Toutes les définitions de tâches qui sont utilisées dans un service doivent avoir la même valeur pour ce paramètre.

Lorsqu'une définition de tâche fait partie d'un service, cette valeur doit correspondre à la valeur `platformFamily` du service.

`cpuArchitecture`

Type : String

Obligatoire : Conditionnelle

Par défaut : X86_64

Si le paramètre est laissé sur `null`, la valeur par défaut est automatiquement attribuée lors du lancement d'une tâche hébergée sur Fargate.

Lorsque vous enregistrez une définition de tâche, vous spécifiez l'architecture du processeur. Les valeurs valides sont `X86_64` et `ARM64`.

Toutes les définitions de tâches qui sont utilisées dans un service doivent avoir la même valeur pour ce paramètre.

Lorsque vous avez des tâches Linux, vous pouvez définir la valeur sur`ARM64`. Pour de plus amples informations, veuillez consulter the section called “Définitions de tâches pour les charges de travail ARM 64 bits”.

Taille de la tâche

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier la quantité totale d'UC et de mémoire utilisée pour cette tâche. Ces valeurs sont distinctes des valeurs `cpu` et `memory` au niveau de la définition de conteneur. Pour les tâches hébergées sur Fargate (Linux et Windows), ces champs sont obligatoires et des valeurs spécifiques sont disponibles selon le `cpu` et la `memory` pris en charge.

Le paramètre suivant est autorisé dans une définition de tâche :

`cpu`

Type : String

Obligatoire : oui

> **Note**
>
> Les paramètres du processeur et de la mémoire au niveau des tâches sont requis et utilisés pour déterminer le type et la taille de l'instance sur laquelle les tâches s'exécutent. Pour les tâches Windows, ces valeurs ne sont pas appliquées au moment de l'exécution, car Windows ne dispose pas d'un mécanisme natif permettant d'appliquer facilement des limites de ressources collectives à un groupe de conteneurs. Si vous souhaitez appliquer des limites de ressources, nous vous recommandons d'utiliser les ressources au niveau du conteneur pour les conteneurs Windows.

Limite stricte du nombre d'unités UC à présenter pour la tâche. Vous pouvez spécifier les valeurs du processeur dans le fichier JSON sous forme de chaîne en unités de processeur ou en mode virtuel CPUs (vCPUs). Par exemple, vous pouvez spécifier une valeur de processeur soit `1024` en unités de processeur, soit `1 vCPU` en CPUs v. Lorsque la définition de tâche est enregistrée, une valeur vCPU est convertie en un entier indiquant les unités du processeur.

Ce champ est obligatoire et vous devez utiliser l'une des valeurs suivantes, qui détermine votre plage de valeurs prises en charge pour le `memory` paramètre. Le tableau suivant indique les combinaisons valides d'UC et de mémoire au niveau de la tâche.

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
256 (0,25 vCPU)	512 Mio, 1 Go, 2 Go	Linux
512 (0,5 vCPU)	1 Go, 2 Go, 3 Go, 4 Go	Linux
1 024 (1 vCPU)	2 Go, 3 Go, 4 Go, 5 Go, 6 Go, 7 Go, 8 Go	Linux, Windows
2 048 (2 vCPU)	Entre 4 Go et 16 Go par incréments de 1 Go	Linux, Windows
4 096 (4 vCPU)	Entre 8 Go et 30 Go par incréments de 1 Go	Linux, Windows

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
8192 (8 vCPU) Note Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 16 Go et 60 Go par incréments de 4 Go	Linux
16384 (16vCPU) Note Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 32 Go et 120 Go par incréments de 8 Go	Linux

`memory`

Type : String

Obligatoire : oui

> **Note**
>
> Les paramètres du processeur et de la mémoire au niveau des tâches sont requis et utilisés pour déterminer le type et la taille de l'instance sur laquelle les tâches s'exécutent. Pour les tâches Windows, ces valeurs ne sont pas appliquées au moment de l'exécution, car Windows ne dispose pas d'un mécanisme natif permettant d'appliquer facilement des limites de ressources collectives à un groupe de conteneurs. Si vous souhaitez appliquer des limites de ressources, nous vous recommandons d'utiliser les ressources au niveau du conteneur pour les conteneurs Windows.

La limite stricte de mémoire à présenter à la tâche. Vous pouvez spécifier des valeurs de mémoire dans la définition de tâche sous forme de chaîne en mégaoctets (MiB) ou en gigaoctets (Go). Par exemple, vous pouvez spécifier une valeur de mémoire `3072` en MiB ou `3 GB` en Go. Lorsque la définition de tâche est enregistrée, la valeur en Go est convertie en un nombre entier indiquant la quantité de Mio.

Ce champ est obligatoire et vous devez utiliser l'une des valeurs suivantes, qui détermine votre plage de valeurs prises en charge pour le `cpu` paramètre :

Valeur de mémoire (en MiB, avec une valeur équivalente approximative en Go)	Valeur d'UC	Systèmes d'exploitation pris en charge pour Fargate
512 (0,5 Go), 1 024 (1 Go), 2 048 (2 Go)	256 (0,25 vCPU)	Linux
1 024 (1 Go), 2 048 (2 Go), 3 072 (3 Go), 4 096 (4 Go)	512 (0,5 vCPU)	Linux
2 048 (2 Go), 3072 (3 Go), 4 096 (4 Go), 5 120 (5 Go), 6 144 (6 Go), 7 168 (7 Go), 8 192 (8 Go)	1 024 (1 vCPU)	Linux, Windows
Entre 4 096 (4 Go) et 16 384 (16 Go) par incréments de 1 024 (1 Go)	2 048 (2 vCPU)	Linux, Windows
Entre 8 192 (8 Go) et 30 720 (30 Go) par incréments de 1 024 (1 Go)	4 096 (4 vCPU)	Linux, Windows
Entre 16 Go et 60 Go par incréments de 4 Go **Note** Cette option nécessite la	8192 (8 vCPU)	Linux

Valeur de mémoire (en MiB, avec une valeur équivalente approximative en Go)	Valeur d'UC	Systèmes d'exploitation pris en charge pour Fargate
plateforme Linux `1.4.0` ou ultérieure		
Entre 32 Go et 120 Go par incréments de 8 Go Note Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	16384 (16vCPU)	Linux

Définitions de conteneur

Lorsque vous enregistrez une définition de tâche, vous devez spécifier une liste de définitions de conteneur qui sont transmises au démon Docker sur une instance de conteneur. Les paramètres suivants sont autorisés dans une définition de conteneur.

Rubriques

- Paramètres de définition de conteneur standards
- Paramètres de définition de conteneur avancés
- Autres paramètres de définition de conteneur

Paramètres de définition de conteneur standards

Les paramètres de définition de tâche suivants sont obligatoires ou utilisés dans la plupart des définitions de conteneur.

Rubriques

- Nom

- Image
- Mémoire
- Mappages de port
- Informations d'identification du référentiel privé

Nom

`name`

Type : String

Obligatoire : oui

Le nom d'un conteneur. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres, des tirets ou des traits de soulignement. Si vous associez plusieurs conteneurs dans une définition de tâche, vous pouvez spécifier l'option `name` d'un conteneur dans l'option `links` d'un autre conteneur. Cela permet de connecter les conteneurs.

Image

`image`

Type : String

Obligatoire : oui

Image utilisée pour démarrer un conteneur. Cette chaîne est transmise directement au démon Docker. Par défaut, les images dans le registre Docker Hub sont disponibles. Vous pouvez également spécifier d'autres référentiels avec soit *`repository-url`*`/`*`image`*`:`*`tag`*, soit *`repository-url`*`/`*`image`*`@`*`digest`*. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres, des tirets, des traits de soulignement, deux points, des points, des barres obliques et des signes dièse. Ce paramètre correspond `Image` à la commande docker create-container et au paramètre `IMAGE` de la commande docker run.

- Lorsqu'une nouvelle tâche démarre, l'agent de conteneur Amazon ECS extrait la version la plus récente de l'image et de l'étiquette spécifiées afin que le conteneur puisse les utiliser. Notez cependant que les mises à jour ultérieures apportées à une image de référentiel ne sont pas répercutées sur les tâches déjà en cours d'exécution.

- Lorsque vous ne spécifiez pas de balise ou de résumé dans le chemin de l'image dans la définition de tâche, l'agent de conteneur Amazon ECS extrait la dernière version de l'image spécifiée.
- Notez cependant que les mises à jour ultérieures apportées à une image de référentiel ne sont pas répercutées sur les tâches déjà en cours d'exécution.
- Les images des registres privés sont prises en charge. Pour de plus amples informations, veuillez consulter Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.
- Les images des référentiels Amazon ECR peuvent être spécifiées en utilisant soit la convention de dénomination complète `registry/repository:tag` ou `registry/repository@digest` (par exemple, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`* ou *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`@`*`sha256:94afd1f2e64d908bc90dbca0035a5b567EXAMPLE`*).
- Les images dans les référentiels officiels sur Docker Hub utilisent un nom unique (par exemple, `ubuntu` ou `mongo`).
- Les images dans les autres référentiels sur Docker Hub sont qualifiées par un nom d'organisation (par exemple, `amazon/amazon-ecs-agent`).
- Les images dans les autres référentiels en ligne sont qualifiées par un nom de domaine (par exemple, `quay.io/assemblyline/ubuntu`).

`versionConsistency`

Type : String

Valeurs valides : `enabled|disabled`

Obligatoire : non

Spécifie si Amazon ECS transformera la balise d'image de conteneur fournie dans la définition du conteneur en un condensé d'image. Par défaut, ce comportement est`enabled`. Si vous définissez la valeur d'un conteneur comme suit`disabled`, Amazon ECS ne transformera pas la balise d'image du conteneur en résumé et utilisera l'URI d'image d'origine spécifiée dans la définition du conteneur pour le déploiement. Pour plus d'informations sur la résolution de l'image du conteneur, consultezRésolution de l'image du conteneur.

Mémoire

`memory`

Type : entier

Obligatoire : non

La quantité de mémoire (en Mio) à présenter au conteneur. Si votre conteneur tente de dépasser la mémoire spécifiée ici, il sera désactivé. La quantité totale de mémoire réservée pour tous les conteneurs au sein d'une tâche doit être inférieure à la valeur `memory` de la tâche, si cette valeur est spécifiée. Ce paramètre correspond `Memory` à la commande docker create-container et à l'option `--memory` docker run.

Le démon Docker 20.10.0 ou ultérieur réserve un minimum de 6 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 6 Mio de mémoire pour vos conteneurs.

Le démon Docker 19.03.13-ce ou antérieur réserve un minimum de 4 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 4 Mio de mémoire pour vos conteneurs.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire d'instance de conteneur Amazon ECS Linux.

`memoryReservation`

Type : entier

Obligatoire : non

La limite flexible (en Mio) de mémoire à réserver pour le conteneur. En cas de contention de la mémoire système, Docker tente de garder la mémoire du conteneur en deçà de cette limite flexible. Toutefois, votre conteneur peut utiliser davantage de mémoire en cas de besoin. Le conteneur peut consommer jusqu'à la limite stricte spécifiée avec le paramètre `memory` (le cas échéant), ou la totalité de la mémoire disponible sur l'instance de conteneur, selon la première valeur atteinte. Ce paramètre correspond `MemoryReservation` à la commande docker create-container et à l'option `--memory-reservation` docker run.

Si aucune valeur de mémoire au niveau de la tâche n'est spécifiée, vous devez indiquer un nombre entier différent de zéro pour `memory` ou `memoryReservation` dans une définition de conteneur. Si vous spécifiez les deux, `memory` doit être supérieur à `memoryReservation`. Si vous spécifiez `memoryReservation`, cette valeur est soustraite des ressources mémoire disponibles pour l'instance de conteneur sur laquelle le conteneur est placé. Sinon, c'est la valeur `memory` qui est utilisée.

Par exemple, supposons que votre conteneur utilise normalement 128 Mio de mémoire, mais qu'il lui arrive d'utiliser jusqu'à 256 Mio de mémoire pendant de courtes périodes. Vous pouvez définir une `memoryReservation` de 128 Mio et une limite stricte `memory` de 300 Mio. Cette configuration permet au conteneur de ne réserver que 128 Mio de mémoire à partir des ressources restantes sur l'instance de conteneur. En même temps, cette configuration permet également au conteneur d'utiliser davantage de ressources mémoire lorsque cela est nécessaire.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

Le démon Docker 20.10.0 ou ultérieur réserve un minimum de 6 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 6 Mio de mémoire pour vos conteneurs.

Le démon Docker 19.03.13-ce ou antérieur réserve un minimum de 4 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 4 Mio de mémoire pour vos conteneurs.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire d'instance de conteneur Amazon ECS Linux.

Mappages de port

`portMappings`

Type : tableau d'objets

Obligatoire : non

Les mappages de ports exposent les ports réseau de votre conteneur au monde extérieur, ce qui permet aux clients d'accéder à votre application. Il est également utilisé pour la communication entre conteneurs dans le cadre d'une même tâche.

Pour les définitions de tâche qui utilisent le mode réseau `awsvpc`, spécifiez uniquement `containerPort`. Le `hostPort` est toujours ignoré et le port du conteneur est automatiquement mappé à un port aléatoire à numéro élevé sur l'hôte.

Les mappages de ports sur Windows utilisent l'adresse de passerelle `NetNAT` plutôt que `localhost`. Il n'existe aucune boucle pour les mappages de ports sous Windows, donc vous ne pouvez pas accéder au port mappé d'un conteneur à partir de l'hôte lui-même.

La plupart des champs de ce paramètre (y compris`containerPort`,`hostPort`,`protocol`) sont mappés `PortBindings` dans la commande docker create-container et dans l'option `--publish` docker run. Si le mode réseau d'une définition de tâche est défini sur `host`, les ports hôtes doivent être indéfinis ou correspondre au port du conteneur dans le mappage de port.

> **Note**
>
> Une fois qu'une tâche passe à l'état `RUNNING`, les affectations manuelles et automatiques de ports de conteneur et d'hôte sont visibles aux emplacements suivants :
>
> - Console : la section Network Bindings (Liaisons réseau) de la description d'un conteneur pour une tâche sélectionnée.
> - AWS CLI : la section `networkBindings` de la sortie de la commande describe-tasks.
> - API : la réponse `DescribeTasks`.
> - Métadonnées : point de terminaison des métadonnées de la tâche.

`appProtocol`

Type : chaîne

Obligatoire : non

Protocole d'application utilisé pour le mappage de port. Ce paramètre s'applique uniquement à Service Connect. Nous vous conseillons de définir ce paramètre de manière cohérente avec le protocole que votre application utilise. Si vous définissez ce paramètre, Amazon ECS ajoute une gestion de connexion spécifique au protocole au proxy Service Connect. Si vous

définissez ce paramètre, Amazon ECS ajoute une télémétrie spécifique au protocole dans la console Amazon ECS et. CloudWatch

Si vous ne définissez aucune valeur pour ce paramètre, le protocole TCP est utilisé. Toutefois, Amazon ECS n'ajoute pas de télémétrie spécifique au protocole pour TCP.

Pour de plus amples informations, veuillez consulter the section called “ Service Connect”.

Valeurs de protocole valides : `"http" | "http2" | "grpc"`

`containerPort`

Type : entier

Obligatoire : oui, lorsque des objets `portMappings` sont utilisés

Le numéro de port sur le conteneur qui est lié au port hôte spécifié par l'utilisateur ou affecté automatiquement.

Pour les tâches utilisant le mode `awsvpc` réseau, vous `containerPort` devez spécifier les ports exposés.

Pour les conteneurs Windows sur Fargate, vous ne pouvez pas utiliser le port 3 150 pour `containerPort`. C'est parce qu'il est réservé.

`containerPortRange`

Type : chaîne

Obligatoire : non

Plage de numéros de port du conteneur liée à la plage de ports hôtes mappés dynamiquement.

Vous ne pouvez définir ce paramètre qu'à l'aide de l'API `register-task-definition`. L'option est disponible dans le paramètre `portMappings`. Pour plus d'informations, consultez register-task-definition dans la référence AWS Command Line Interface .

Les règles suivantes s'appliquent lorsque vous spécifiez une `containerPortRange` :

- Vous devez utiliser le mode `awsvpc` réseau.
- Ce paramètre est disponible pour les systèmes d'exploitation Windows et Linux.

- L'instance de conteneur doit au moins disposer de la version 1.67.0 de l'agent de conteneur et au moins de la version 1.67.0-1 du package `ecs-init`.
- Vous pouvez spécifier un maximum de 100 plages de ports pour chaque conteneur.
- Vous ne spécifiez pas de `hostPortRange`. La valeur de `hostPortRange` est définie comme suit :
  - Pour les conteneurs d'une tâche en mode réseau `awsvpc`, le `hostPort` est défini sur la même valeur que le `containerPort`. Il s'agit d'une stratégie de mappage statique.
- Les valeurs valides de `containerPortRange` sont comprises entre 1 et 65535.
- Un port peut uniquement être inclus dans un mappage de port pour chaque conteneur.
- Vous ne pouvez pas spécifier de plages de ports qui se chevauchent.
- Le premier port de la plage doit être inférieur au dernier port de la plage.
- Docker vous recommande de désactiver le proxy Docker dans le fichier de configuration du démon Docker en présence d'un grand nombre de ports.

  Pour plus d'informations, consultez le numéro #11185 sur GitHub.

  Pour plus d'informations sur la désactivation du proxy Docker dans le fichier de configuration du démon Docker, veuillez consulter Démon Docker dans le Guide du développeur Amazon ECS (langue française non garantie).

Vous pouvez appeler DescribeTasks pour voir la `hostPortRange` correspondant aux ports hôtes liés aux ports de conteneur.

Les plages de ports ne sont pas incluses dans les événements de tâches Amazon ECS, qui sont envoyés à EventBridge. Pour de plus amples informations, veuillez consulter the section called “Automatisez les réponses aux erreurs Amazon ECS à l'aide de EventBridge”.

`hostPortRange`

Type : chaîne

Obligatoire : non

Plage de numéros de port sur l'hôte utilisée avec la liaison réseau. Elle est attribuée par Docker et délivrée par l'agent Amazon ECS.

`hostPort`

Type : entier

Obligatoire : non

Le numéro de port sur l'instance de conteneur à réserver pour votre conteneur.

Le champ `hostPort` peut être laissé vide ou avoir la même valeur que`containerPort`.

La plage de ports éphémères par défaut pour Docker 1.6.0 et versions ultérieures est répertoriée dans l'instance sous `/proc/sys/net/ipv4/ip_local_port_range`. Si ce paramètre de noyau n'est pas disponible, la plage de ports éphémères par défaut de `49153-65535` est utilisée. N'essayez pas de spécifier un port d'hôte dans la plage de ports éphémères. Cela est dû au fait qu'ils sont réservés à une affectation automatique. En général, les ports inférieurs à `32768` ne sont pas compris dans la plage de ports éphémères.

Les ports réservés par défaut sont le port `22` pour SSH, les ports Docker `2375` et `2376`, et les ports d'agent de conteneur Amazon ECS `51678-51680`. Tout port hôte ayant été spécifié précédemment par l'utilisateur pour une tâche en cours d'exécution est également réservé pendant l'exécution de la tâche. Après l'arrêt d'une tâche, le port hôte est libéré. Les ports réservés actuels s'affichent dans le champ `remainingResources` de la sortie describe-container-instances. Une instance de conteneur peut avoir jusqu'à 100 ports réservés à la fois, y compris les ports réservés par défaut. Les ports affectés automatiquement ne sont pas pris en compte dans le quota des 100 ports réservés.

`name`

Type : String

Obligatoire : Non, obligatoire pour que Service Connect et VPC Lattice soient configurés dans un service

Nom utilisé pour le mappage de port. Ce paramètre s'applique uniquement à Service Connect et à VPC Lattice. Ce paramètre est le nom que vous utilisez dans la configuration Service Connect et VPC Lattice d'un service.

Pour de plus amples informations, veuillez consulter Utilisez Service Connect pour connecter les services Amazon ECS avec des noms abrégés.

Dans l'exemple suivant, les deux champs obligatoires pour Service Connect et VPC Lattice sont utilisés.

```
"portMappings": [
    {
        "name": string,
        "containerPort": integer
    }
]
```

`protocol`

Type : chaîne

Obligatoire : non

Le protocole utilisé pour le mappage de port. Les valeurs valides sont `tcp` et `udp`. L'argument par défaut est `tcp`.

> **Important**
>
> Seul `tcp` est pris en charge pour Service Connect. N'oubliez pas que `tcp` est implicite si ce champ n'est pas défini.

Si vous spécifiez un port hôte, utilisez la syntaxe suivante.

```
"portMappings": [
    {
        "containerPort": integer,
        "hostPort": integer
    }
    ...
]
```

Si vous souhaitez utiliser un port hôte affecté automatiquement, utilisez la syntaxe suivante.

```
"portMappings": [
    {
        "containerPort": integer
    }
    ...
]
```

Informations d'identification du référentiel privé

`repositoryCredentials`

Type : objet RepositoryCredentials

Obligatoire : non

Informations d'identification du référentiel pour l'authentification de registre privé.

Pour de plus amples informations, veuillez consulter Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.

`credentialsParameter`

Type : String

Obligatoire : oui, lorsque des objets `repositoryCredentials` sont utilisés

L'Amazon Resource Name (ARN) du secret contenant les informations d'identification du référentiel privé.

Pour de plus amples informations, veuillez consulter Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.

> **Note**
>
> Lorsque vous utilisez l'API Amazon ECS AWS CLI AWS SDKs, ou si le secret existe dans la même région que la tâche que vous lancez, vous pouvez utiliser l'ARN complet ou le nom du secret. Lorsque vous utilisez le AWS Management Console, vous devez spécifier l'ARN complet du secret.

Voici un extrait de code d'une définition de tâche indiquant les paramètres requis :

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
```

```
        }
    }
]
```

Paramètres de définition de conteneur avancés

Les paramètres de définition de conteneur avancés suivants fournissent des fonctionnalités étendues à la commande docker run utilisée pour lancer des conteneurs sur vos instances de conteneur Amazon ECS.

Rubriques

- Politique de redémarrage
- Surveillance de l'état
- Environnement
- Paramètres réseau
- Stockage et journalisation
- Sécurité
- Limites des ressources
- Étiquettes Docker

Politique de redémarrage

`restartPolicy`

La politique de redémarrage du conteneur et les paramètres de configuration associés. Lorsque vous configurez une politique de redémarrage pour un conteneur, Amazon ECS peut redémarrer le conteneur sans avoir à remplacer la tâche. Pour de plus amples informations, veuillez consulter Redémarrer des conteneurs individuels dans des tâches Amazon ECS avec des politiques de redémarrage de conteneurs.

`enabled`

Type : booléen

Obligatoire : oui

Spécifie si une politique de redémarrage est activée pour le conteneur.

`ignoredExitCodes`

Type : tableau integer

Obligatoire : non

Liste des codes de sortie qu'Amazon ECS ignorera et ne tentera pas de redémarrer. Vous pouvez spécifier un maximum de 50 codes de sortie de conteneur. Par défaut, Amazon ECS n'ignore aucun code de sortie.

`restartAttemptPeriod`

Type : entier

Obligatoire : non

Période (en secondes) pendant laquelle le conteneur doit fonctionner avant de pouvoir tenter de redémarrer. Un conteneur ne peut être redémarré qu'une fois par `restartAttemptPeriod` seconde. Si un conteneur n'est pas en mesure de fonctionner pendant cette période et qu'il sort plus tôt que prévu, il ne sera pas redémarré. Vous pouvez définir un minimum `restartAttemptPeriod` de 60 secondes et un maximum `restartAttemptPeriod` de 1800 secondes. Par défaut, un conteneur doit fonctionner pendant 300 secondes avant de pouvoir être redémarré.

Surveillance de l'état

`healthCheck`

Commande de surveillance de l'état du conteneur et paramètres de configuration associés pour le conteneur. Pour de plus amples informations, veuillez consulter Déterminer l'état des tâches Amazon ECS à l'aide de vérifications de l'état des conteneurs.

`command`

Tableau de chaînes représentant la commande que le conteneur exécute pour déterminer si celle-ci est saine. Le tableau de chaînes peut commencer par `CMD` pour exécuter directement les arguments de la commande, ou par `CMD-SHELL` pour exécuter la commande avec le shell par défaut du conteneur. Si vous n'en spécifiez aucun, `CMD` est utilisé.

Lorsque vous enregistrez une définition de tâche dans le AWS Management Console, utilisez une liste de commandes séparées par des virgules. Ces commandes sont converties en une

chaîne une fois la définition de tâche créée. Un exemple d'entrée pour une surveillance de l'état est le suivant.

```
CMD-SHELL, curl -f http://localhost/ || exit 1
```

Lorsque vous enregistrez une définition de tâche à l'aide du AWS CLI panneau AWS Management Console JSON APIs, placez la liste des commandes entre crochets. Un exemple d'entrée pour une surveillance de l'état est le suivant.

```
[ "CMD-SHELL", "curl -f http://localhost/ || exit 1" ]
```

Un code de sortie de 0, avec aucune sortie `stderr`, indique la réussite et un code de sortie autre que zéro indique qu'il s'agit d'un échec.

`interval`

La durée (en secondes) entre chaque surveillance de l'état. Vous pouvez indiquer une durée comprise entre 5 et 300 secondes. La valeur par défaut est de 30 secondes.

`timeout`

La durée (en secondes) d'attente pour qu'une surveillance de l'état réussisse avant qu'elle ne soit considérée comme un échec. Vous pouvez indiquer une durée comprise entre 2 et 60 secondes. La valeur par défaut est de 5 secondes.

`retries`

Nombre de nouvelles tentatives d'exécution d'une surveillance de l'état ayant échoué avant que le conteneur soit considéré comme défectueux. Vous pouvez indiquer un nombre de tentatives compris en 1 et 10. La valeur par défaut est 3 nouvelles tentatives.

`startPeriod`

La période de grâce facultative pour donner aux conteneurs le temps de démarrer avant l'échec des surveillance de l'état est prise en compte dans le nombre maximal de tentatives. Vous pouvez spécifier une valeur comprise entre 0 et 300 secondes. Par défaut, la `startPeriod` est désactivée.

Si une vérification de l'état aboutit dans le délai défini par `startPeriod`, le conteneur est considéré comme sain et toutes les défaillances suivantes sont prises en compte dans le nombre maximal de nouvelles tentatives.

Environnement

`cpu`

Type : entier

Obligatoire : non

Le nombre d'unités `cpu` que l'agent de conteneur Amazon ECS réserve pour le conteneur. Sous Linux, ce paramètre correspond `CpuShares` à la section Créer un conteneur.

Ce champ est facultatif pour les tâches qui utilisent le type de lancement Fargate. La quantité totale de processeurs réservée pour tous les conteneurs au sein d'une tâche doit être inférieure à la valeur `cpu` au niveau de la tâche.

Les conteneurs Linux partagent des unités de processeur non allouées avec les autres conteneurs de l'instance de conteneur selon le même ratio que la quantité allouée. Par exemple, supposons que vous exécutez une tâche à conteneur unique sur un type d'instance à cœur unique avec 512 unités de processeur spécifiées pour ce conteneur. De plus, cette tâche est la seule tâche en cours d'exécution sur l'instance de conteneur. Dans cet exemple, le conteneur peut utiliser le partage complet de 1 024 unités UC à tout moment. Toutefois, supposons que vous ayez lancé une autre copie de la même tâche sur cette instance de conteneur. Un minimum de 512 unités de processeur est affecté à chaque tâche si nécessaire. De même, si l'autre conteneur n'utilise pas le processeur restant, chaque conteneur peut bénéficier d'une utilisation plus importante du processeur. Toutefois, si les deux tâches sont actives à 100 % en permanence, elles sont limitées à 512 unités d'UC.

Sur les instances de conteneur Linux, le démon Docker de l'instance de conteneur se sert de la valeur de processeur pour calculer les ratios de partage de processeur relatifs pour les conteneurs en cours d'exécution. La valeur de partage de processeur valide minimale autorisée par le noyau Linux est de 2, et la valeur maximale de partage de processeur valide autorisée par le noyau Linux est de 262 144. Cependant, le paramètre CPU n'est pas obligatoire et vous pouvez utiliser des valeurs de processeur inférieures à deux et supérieures à 262144 dans vos définitions de conteneur. Pour les valeurs de processeur inférieures à deux (y compris null) et supérieures à 262144, le comportement varie en fonction de la version de votre agent de conteneur Amazon ECS :

Sur les instances de conteneur Windows, le quota de processeur est appliqué comme quota absolu. Les conteneurs Windows n'ont accès qu'à la quantité spécifiée de processeur décrite

dans la définition de tâche. Une valeur de processeur nulle ou égale à zéro est transmise à Docker comme étant `0`. Windows interprète ensuite cette valeur comme 1 % d'un processeur.

Pour voir d'autres exemples, veuillez consulter le billet de blog How Amazon ECS manages CPU and memory resources.

`gpu`

Ce paramètre n'est pas pris en charge pour les conteneurs hébergés sur Fargate.

Type : objet ResourceRequirement

Obligatoire : non

Le nombre de `GPUs` physiques que l'agent de conteneur Amazon ECS réserve pour le conteneur. Le nombre de conteneurs GPUs réservés à tous les conteneurs d'une tâche ne doit pas dépasser le nombre de conteneurs disponibles GPUs sur l'instance de conteneur sur laquelle la tâche est lancée. Pour de plus amples informations, veuillez consulter Définitions de tâches Amazon ECS pour les charges de travail du GPU.

`Elastic Inference accelerator`

Ce paramètre n'est pas pris en charge pour les conteneurs hébergés sur Fargate.

Type : objet ResourceRequirement

Obligatoire : non

Pour le type `InferenceAccelerator`, `value` correspond à `deviceName` d'un `InferenceAccelerator` spécifié dans une définition de tâche. Pour de plus amples informations, veuillez consulter the section called “Nom de l'accélérateur Elastic Inference”.

`essential`

Type : booléen

Obligatoire : non

Supposons que le paramètre `essential` d'un conteneur soit marqué comme `true`, et que ce conteneur échoue ou s'arrête pour une raison quelconque. Ensuite, tous les autres conteneurs qui font partie de la tâche sont arrêtés. Si le paramètre `essential` d'un conteneur a la valeur `false`, son échec n'a pas d'incidence sur le reste des conteneurs dans la tâche. Si ce paramètre n'est pas spécifié, le conteneur est supposé essentiel.

Toutes les tâches doivent comporter au moins un conteneur essentiel. Supposons que vous disposiez d'une application composée de plusieurs conteneurs. Ensuite, les conteneurs de groupes qui sont utilisés dans un même but pour former des composants, et séparer les différents composants en plusieurs définitions de tâches. Pour de plus amples informations, veuillez consulter Architectez votre application pour Amazon ECS.

```
"essential": true|false
```

`entryPoint`

> **Important**
>
> Les premières versions de l'agent de conteneur Amazon ECS ne traitent pas correctement les paramètres `entryPoint`. Si vous rencontrez des difficultés pour utiliser `entryPoint`, mettez à jour l'agent de conteneur ou saisissez plutôt vos commandes et vos arguments sous forme d'éléments de tableau `command`.

Type : tableau de chaînes

Obligatoire : non

Point d'entrée qui est transmis au conteneur.

```
"entryPoint": ["string", ...]
```

`command`

Type : tableau de chaînes

Obligatoire : non

La commande transmise au conteneur. Ce paramètre correspond `Cmd` à la commande create-container et au `COMMAND` paramètre à docker run. S'il existe plusieurs arguments, assurez-vous que chaque argument est une chaîne séparée dans le tableau.

```
"command": ["string", ...]
```

`workingDirectory`

Type : chaîne

Obligatoire : non

Répertoire de travail dans lequel exécuter les commandes à l'intérieur du conteneur. Ce paramètre correspond à `WorkingDir` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--workdir` correspond à docker run.

```
"workingDirectory": "string"
```

`environmentFiles`

Cela n'est pas disponible pour les conteneurs Windows sur Fargate.

Type : tableau d'objets

Obligatoire : non

Liste des fichiers contenant les variables d'environnement à transmettre à un conteneur. Ce paramètre correspond à l'`--env-file`option de la commande docker run.

Vous pouvez spécifier jusqu'à dix fichiers d'environnement. Le fichier doit avoir une extension de fichier `.env`. Chaque ligne d'un fichier d'environnement contient une variable d'environnement au format `VARIABLE=VALUE`. Les lignes commençant par # sont traitées comme des commentaires et sont ignorées.

Si des variables d'environnement individuelles sont spécifiées dans la définition du conteneur, elles ont priorité sur les variables contenues dans un fichier d'environnement. Si plusieurs fichiers d'environnement contenant la même variable sont spécifiés, ils sont traités de haut en bas. Nous vous recommandons d'utiliser des noms de variables uniques. Pour de plus amples informations, veuillez consulter Transmettre une variable d'environnement individuelle à un conteneur Amazon ECS.

`value`

Type : String

Obligatoire : oui

L'Amazon Resource Name (ARN) de l'objet Amazon S3 contenant le fichier de variable d'environnement.

`type`

Type : String

Obligatoire : oui

Type de fichier à utiliser La seule valeur prise en charge est `s3`.

`environment`

Type : tableau d'objets

Obligatoire : non

Variables d'environnement à transmettre à un conteneur. Ce paramètre correspond `Env` à la commande docker create-container et à l'option `--env` à la commande docker run.

> **Important**
>
> Nous déconseillons l'utilisation des variables d'environnement en texte clair pour les informations sensibles, comme les données d'identification.

`name`

Type : String

Obligatoire : oui lorsque `environment` est utilisé

Nom de la variable d'environnement.

`value`

Type : String

Obligatoire : oui lorsque `environment` est utilisé

Valeur de la variable d'environnement.

```
"environment" : [
    { "name" : "string", "value" : "string" },
    { "name" : "string", "value" : "string" }
]
```

`secrets`

Type : tableau d'objets

Obligatoire : non

Objet représentant le secret à exposer à votre conteneur. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.

`name`

Type : String

Obligatoire : oui

Valeur à définir comme variable d'environnement sur le conteneur.

`valueFrom`

Type : String

Obligatoire : oui

Secret à exposer au conteneur. Les valeurs prises en charge sont soit le nom Amazon Resource (ARN) complet du AWS Secrets Manager secret, soit l'ARN complet du paramètre dans le AWS Systems Manager Parameter Store.

> **Note**
>
> Si le paramètre Systems Manager Parameter Store ou le paramètre Secrets Manager existe au même endroit Région AWS que la tâche que vous lancez, vous pouvez utiliser l'ARN complet ou le nom du secret. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

```
"secrets": [
    {
        "name": "environment_variable_name",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }
]
```

Paramètres réseau

`disableNetworking`

Ce paramètre n'est pas pris en charge pour les tâches exécutées sur Fargate.

Type : booléen

Obligatoire : non

Quand ce paramètre a la valeur true, la mise en réseau est désactivée dans le conteneur.

L'argument par défaut est `false`.

```
"disableNetworking": true|false
```

`links`

Ce paramètre n'est pas pris en charge pour les tâches utilisant le mode `awsvpc` réseau.

Type : tableau de chaînes

Obligatoire : non

Le paramètre `link` permet aux conteneurs de communiquer entre eux sans avoir besoin de définir des mappages de ports. Ce paramètre n'est pris en charge que si le mode réseau d'une définition de tâche est défini sur `bridge`. La construction `name:internalName` est analogue à `name:alias` dans les liaisons Docker. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits d'union et traits de soulignement sont autorisés.

> **Important**
>
> Des conteneurs situés sur la même instance de conteneur peuvent communiquer entre eux sans nécessiter de liens ou de mappages de ports hôtes. L'isolation réseau sur une instance de conteneur est contrôlée par les groupes de sécurité et les paramètres de VPC.

```
"links": ["name:internalName", ...]
```

`hostname`

Type : chaîne

Obligatoire : non

Nom d'hôte à utiliser pour votre conteneur. Ce paramètre correspond `Hostname` au docker create-container et à l'option `--hostname` docker run.

> **Note**
>
> Si vous utilisez le mode réseau `awsvpc`, le paramètre `hostname` n'est pas pris en charge.

```
"hostname": "string"
```

`dnsServers`

Cela n'est pas pris en charge pour les tâches exécutées sur Fargate.

Type : tableau de chaînes

Obligatoire : non

Liste de serveurs DNS qui sont présentés au conteneur.

```
"dnsServers": ["string", ...]
```

`extraHosts`

Ce paramètre n'est pas pris en charge pour les tâches utilisant le mode `awsvpc` réseau.

Type : tableau d'objets

Obligatoire : non

Liste de noms d'hôte et de mappages d'adresses IP à ajouter au fichier `/etc/hosts` dans le conteneur.

Ce paramètre correspond `ExtraHosts` à la commande docker create-container et à l'option `--add-host` docker run.

```
"extraHosts": [
      {
        "hostname": "string",
```

```
        "ipAddress": "string"
      }
      ...
    ]
```

`hostname`

Type : String

Obligatoire : oui, lorsque des objets `extraHosts` sont utilisés

Nom d'hôte à utiliser dans l'entrée `/etc/hosts`.

`ipAddress`

Type : String

Obligatoire : oui, lorsque des objets `extraHosts` sont utilisés

Adresse IP à utiliser dans l'entrée `/etc/hosts`.

Stockage et journalisation

`readonlyRootFilesystem`

Type : booléen

Obligatoire : non

Quand ce paramètre a la valeur true, le conteneur ne dispose que d'un accès en lecture seule au système de fichiers racine. Ce paramètre correspond à l'option `--read-only` docker run `ReadonlyRootfs` dans la commande docker create-container.

> **Note**
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

L'argument par défaut est `false`.

```
"readonlyRootFilesystem": true|false
```

`mountPoints`

Type : tableau d'objets

Obligatoire : non

Les points de montage des volumes de données de votre conteneur. Ce paramètre correspond `Volumes` à l'API Docker create-container et `--volume` à l'option docker run.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`. Les conteneurs Windows ne peuvent pas monter de répertoires sur un autre lecteur, et les points de montage ne peuvent pas être utilisés sur plusieurs lecteurs. Vous devez spécifier des points de montage pour associer un volume Amazon EBS directement à une tâche Amazon ECS.

`sourceVolume`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Le chemin dans le conteneur où le volume sera monté.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

Pour les tâches exécutées sur EC2 des instances exécutant le système d'exploitation Windows, laissez la valeur par défaut de`false`.

`volumesFrom`

Type : tableau d'objets

Obligatoire : non

Volumes de données à monter à partir d'un autre conteneur. Ce paramètre correspond `VolumesFrom` à la commande docker create-container et à l'option `--volumes-from` docker run.

`sourceContainer`

Type : String

Obligatoire : oui lorsque `volumesFrom` est utilisé

Nom du conteneur à partir duquel monter les volumes.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

```
"volumesFrom": [
                {
                  "sourceContainer": "string",
                  "readOnly": true|false
                }
              ]
```

`logConfiguration`

Type : LogConfigurationObjet

Obligatoire : non

Spécification de configuration du journal pour le conteneur.

Pour obtenir des exemples de définitions de tâche qui utilisent une configuration de journaux, consultez Exemples de définitions de tâches Amazon ECS.

Ce paramètre correspond `LogConfig` à la commande docker create-container et à l'option `--log-driver` docker run. Par défaut, les conteneurs utilisent le même pilote de journalisation que celui utilisé par le démon Docker. Cependant, le conteneur peut utiliser un pilote de journalisation différent que celui du démon Docker en spécifiant un pilote de journal avec ce paramètre dans la définition de conteneur. Pour utiliser un pilote de journalisation différent pour un conteneur, le système de journal doit être configuré correctement sur l'instance de conteneur (ou sur un serveur de journal différent pour les options de journalisation à distance).

Tenez compte des éléments suivants lorsque vous spécifiez une configuration de journal pour vos conteneurs :

- Amazon ECS prend en charge un sous-ensemble des pilotes de journalisation qui sont disponibles pour le démon Docker. Les pilotes de journal supplémentaires qui ne sont pas pris en charge actuellement seront peut-être disponibles dans les prochaines versions de l'agent de conteneur Amazon ECS.
- Ce paramètre nécessite la version 1.18 ou ultérieure de l'API Docker à distance sur votre instance de conteneur.
- Vous devez installer tout logiciel supplémentaire en dehors de cette tâche. Par exemple, les agrégateurs de sortie Fluentd ou un hôte distant exécutant Logstash auquel envoyer des journaux Gelf.

```
"logConfiguration": {
      "logDriver": "awslogs","fluentd","gelf","json-
file","journald","splunk","syslog","awsfirelens",
      "options": {"string": "string"
        ...},
 "secretOptions": [{
  "name": "string",
  "valueFrom": "string"
 }]
}
```

`logDriver`

Type : String

Valeurs valides : `"awslogs","fluentd","gelf","json-file","journald","splunk","syslog","awsfirelens"`

Obligatoire : oui lorsque `logConfiguration` est utilisé

Pilote de journal à utiliser pour le conteneur. Par défaut, les valeurs valides mentionnées ci-dessus sont les pilotes de journal avec lesquels l'agent de conteneur Amazon ECS peut communiquer.

Les pilotes de journal pris en charge sont `awslogssplunk`, et`awsfirelens`.

Pour plus d'informations sur l'utilisation du pilote de `awslogs` journal dans les définitions de tâches pour envoyer les journaux de vos conteneurs à CloudWatch Logs, consultezEnvoyez les journaux Amazon ECS à CloudWatch .

Pour de plus amples informations sur l'utilisation du pilote de journal `awsfirelens`, consultez Routage personnalisé des journaux.

> **Note**
>
> Si vous avez un pilote personnalisé qui n'est pas répertorié, vous pouvez bifurquer le projet d'agent de conteneur Amazon ECS disponible sur GitHub et le personnaliser pour qu'il fonctionne avec ce pilote. Nous vous conseillons d'envoyer des demandes d'extraction pour les modifications que vous souhaitez inclure. Toutefois, nous ne prenons pas en charge l'exécution de copies modifiées de ce logiciel pour le moment.

Ce paramètre nécessite la version 1.18 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

`options`

Type : mappage chaîne/chaîne

Obligatoire : non

key/value Carte des options de configuration à envoyer au pilote du journal.

Les options que vous pouvez spécifier dépendent du pilote de journal. Certaines des options que vous pouvez spécifier lorsque vous utilisez le `awslogs` routeur pour acheminer les journaux vers Amazon CloudWatch sont les suivantes :

`awslogs-create-group`

Obligatoire : non

Spécifiez si vous souhaitez que le groupe de journaux soit créé automatiquement. Si cette option n'est pas spécifiée, l'emplacement par défaut est `false`.

> **Note**
>
> Votre politique IAM doit inclure l'autorisation `logs:CreateLogGroup` afin que vous puissiez essayer d'utiliser `awslogs-create-group`.

`awslogs-region`

Obligatoire : oui

Spécifiez Région AWS le pilote de `awslogs` journal auquel envoyer vos journaux Docker. Vous pouvez choisir d'envoyer tous vos journaux provenant de clusters situés dans différentes régions vers une seule région dans CloudWatch Logs. Cela leur permet d'être tous visibles en un seul endroit. Sinon, vous pouvez les séparer par région pour plus de granularité. Assurez-vous que le groupe de journaux spécifié existe dans la région que vous spécifiez avec cette option.

`awslogs-group`

Obligatoire : oui

Assurez-vous de spécifier un groupe de journaux auquel le pilote de journaux `awslogs` envoie ses flux de journaux.

`awslogs-stream-prefix`

Obligatoire : oui

Utilisez l'option `awslogs-stream-prefix` pour associer un flux de journaux au préfixe spécifié, au nom du conteneur et à l'ID de la tâche Amazon ECS à laquelle appartient le conteneur. Si vous spécifiez un préfixe avec cette option, le format du flux de journaux est le suivant.

```
prefix-name/container-name/ecs-task-id
```

Si vous ne spécifiez pas de préfixe avec cette option, le flux de journaux est nommé d'après l'ID de conteneur attribué par le démon Docker sur l'instance de conteneur. Etant

donné qu'il est difficile de suivre les journaux jusqu'au conteneur qui les a envoyé en utilisant uniquement l'ID de conteneur Docker (qui est uniquement disponible sur l'instance de conteneur), nous vous recommandons de spécifier un préfixe avec cette option.

Pour les services Amazon ECS, vous pouvez utiliser le nom du service comme préfixe. Ce faisant, vous pourrez suivre les flux de journaux et déterminer le service auquel appartient le conteneur, trouver le nom du conteneur qui les a envoyés, ainsi que l'ID de la tâche à laquelle le conteneur appartient.

Vous devez spécifier un préfixe de flux pour vos journaux pour qu'ils apparaissent dans le panneau des journaux lors de l'utilisation de la console Amazon ECS.

`awslogs-datetime-format`

Obligatoire : non

Cette option définit un modèle de démarrage à plusieurs lignes au format `strftime` Python. Un message de journal est composé de plusieurs lignes : une ligne correspondant au modèle et les lignes suivantes qui ne correspondent pas au modèle. La ligne mise en correspondance est le délimiteur entre les messages de journalisation.

Ce format peut, par exemple, servir à analyser une sortie comme une pile de vidage, laquelle pourrait, dans le cas contraire, être consignée en plusieurs entrées. Le modèle adéquat permet de la capturer dans une seule entrée.

Pour de plus amples informations, veuillez consulter awslogs-datetime-format.

Vous ne pouvez pas configurer à la fois les options `awslogs-datetime-format` et `awslogs-multiline-pattern`.

> **Note**
>
> La journalisation multiligne effectue l'analyse et la mise en correspondance des expressions régulières de tous les messages de journalisation. Cela peut avoir un impact négatif sur les performances de journalisation.

`awslogs-multiline-pattern`

Obligatoire : non

Cette option définit un modèle de démarrage à plusieurs lignes à l'aide d'une expression régulière. Un message de journal est composé de plusieurs lignes : une ligne correspondant au modèle et les lignes suivantes qui ne correspondent pas au modèle. La ligne mise en correspondance est le délimiteur entre les messages de journalisation.

Pour de plus amples informations, veuillez consulter awslogs-multiline-pattern.

Cette option est ignorée si `awslogs-datetime-format` est également configuré.

Vous ne pouvez pas configurer à la fois les options `awslogs-datetime-format` et `awslogs-multiline-pattern`.

> **Note**
>
> La journalisation multiligne effectue l'analyse et la mise en correspondance des expressions régulières de tous les messages de journalisation. Cela peut avoir un impact négatif sur les performances de journalisation.

Les options suivantes s'appliquent à tous les pilotes de journal pris en charge.

`mode`

Obligatoire : non

Valeurs valides : `non-blocking` | `blocking`

Cette option définit le mode de livraison des messages de journal du conteneur au pilote de journal spécifié à l'aide de`logDriver`. Le mode de livraison que vous choisissez affecte la disponibilité des applications lorsque le flux de logs provenant du conteneur est interrompu.

Si vous utilisez ce `blocking` mode et que le flux de logs est interrompu, les appels provenant du code du conteneur pour écrire dans les `stderr` flux `stdout` et seront bloqués. Le fil de journalisation de l'application se bloquera en conséquence. Cela peut empêcher l'application de répondre et entraîner l'échec de vérification de l'état du conteneur.

Si vous utilisez le mode `non-blocking`, les journaux du conteneur sont plutôt stockés dans une mémoire tampon intermédiaire configurée avec l'option `max-buffer-size`. Cela empêche l'application de ne plus répondre lorsque les journaux ne peuvent pas

être envoyés. Nous vous recommandons d'utiliser ce mode si vous souhaitez garantir la disponibilité du service et si vous acceptez une certaine perte de journaux. Pour plus d'informations, consultez la section Prévention de la perte de journal grâce au mode non bloquant dans le pilote de journal du `awslogs` conteneur.

Vous pouvez définir une valeur par défaut `mode` pour tous les conteneurs d'un conteneur spécifique en Région AWS utilisant les paramètres du `defaultLogDriverMode` compte. Si vous ne spécifiez pas l'`mode`option dans les paramètres du compte `logConfiguration` ou si vous ne configurez pas le paramètre, Amazon ECS passera en `non-blocking` mode par défaut. Pour plus d'informations sur le paramétrage du compte, consultezMode pilote de journal par défaut.

Quand le mode `non-blocking` est utilisé, l'option de journalisation `max-buffer-size` contrôle la taille de la mémoire tampon utilisée pour le stockage des messages intermédiaires. Assurez-vous de spécifier une taille de mémoire tampon adéquate en fonction de votre application. La quantité totale de mémoire allouée au niveau de la tâche doit être supérieure à la quantité de mémoire allouée à tous les conteneurs en plus de la mémoire tampon du pilote de journal.

> **Note**
>
> Le 25 juin 2025, Amazon ECS a modifié le mode pilote de journal par défaut de à pour donner la priorité `blocking` `non-blocking` à la disponibilité des tâches plutôt qu'à la journalisation. Pour continuer à utiliser le `blocking` mode après cette modification, effectuez l'une des opérations suivantes :
>
> - Définissez l'`mode`option dans votre définition de conteneur `logConfiguration` comme`blocking`.
> - Réglez le paramètre du `defaultLogDriverMode` compte sur`blocking`.

`max-buffer-size`

Obligatoire : non

Valeur par défaut : `10m`

Quand le mode `non-blocking` est utilisé, l'option de journalisation `max-buffer-size` contrôle la taille de la mémoire tampon utilisée pour le stockage des messages

intermédiaires. Assurez-vous de spécifier une taille de mémoire tampon adéquate en fonction de votre application. Lorsque la mémoire tampon est pleine, il est impossible de stocker d'autres journaux. Les journaux qui ne peuvent pas être stockés sont perdus.

Pour acheminer les journaux à l'aide du `splunk` routeur de journaux, vous devez spécifier a `splunk-token` et `splunk-url` a.

Lorsque vous utilisez le routeur de `awsfirelens` journaux pour acheminer les journaux vers une AWS Partner Network destination Service AWS ou à des fins de stockage et d'analyse des journaux, vous pouvez définir l'`log-driver-buffer-limit`option permettant de limiter le nombre d'événements mis en mémoire tampon avant d'être envoyés au conteneur du routeur de journaux. Cela peut aider à résoudre le problème potentiel de perte de journal, car un débit élevé peut entraîner un épuisement de la mémoire pour le tampon à l'intérieur de Docker. Pour de plus amples informations, veuillez consulter the section called “Configuration des journaux pour un débit élevé”.

Les autres options que vous pouvez spécifier lors de l'utilisation `awsfirelens` pour acheminer les journaux dépendent de la destination. Lorsque vous exportez des journaux vers Amazon Data Firehose, vous pouvez spécifier le Région AWS format `region` et le nom du flux de journaux. `delivery_stream`

Lorsque vous exportez des journaux vers Amazon Kinesis Data Streams, vous pouvez spécifier Région AWS un `region` with et un nom `stream` de flux de données avec.

Lorsque vous exportez des journaux vers Amazon OpenSearch Service, vous pouvez spécifier des options telles que `Name` `Host` (point de terminaison de OpenSearch service sans protocole) `Port` `IndexType`,`Aws_auth`,`Aws_region`,,`Suppress_Type_Name`, et`tls`.

Lorsque vous exportez des journaux vers Amazon S3, vous pouvez spécifier le compartiment à l'aide de l'`bucket`option. Vous pouvez également spécifier`region`, `total_file_size``upload_timeout`, et `use_put_object` en tant qu'options.

Ce paramètre nécessite la version 1.19 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

`secretOptions`

Type : tableau d'objets

Obligatoire : non

Objet qui représente le secret à transmettre à la configuration de journal. Les secrets utilisés dans la configuration du journal peuvent inclure un jeton d'authentification, un certificat ou une clé de chiffrement. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.

`name`

Type : String

Obligatoire : oui

Valeur à définir comme variable d'environnement sur le conteneur.

`valueFrom`

Type : String

Obligatoire : oui

Secret à exposer à la configuration de journal du conteneur.

```
"logConfiguration": {
 "logDriver": "splunk",
 "options": {
  "splunk-url": "https://cloud.splunk.com:8080",
  "splunk-token": "...",
  "tag": "...",
  ...
 },
 "secretOptions": [{
  "name": "splunk-token",
  "valueFrom": "/ecs/logconfig/splunkcred"
 }]
}
```

`firelensConfiguration`

Type : FirelensConfigurationObjet

Obligatoire : non

La FireLens configuration du conteneur. Cette option est utilisée pour spécifier et configurer un routeur de journal pour les journaux de conteneur. Pour de plus amples informations, veuillez consulter Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

```
{
    "firelensConfiguration": {
        "type": "fluentd",
        "options": {
            "KeyName": ""
        }
    }
}
```

`options`

Type : mappage chaîne/chaîne

Obligatoire : non

key/value Carte des options à utiliser lors de la configuration du routeur de journalisation. Ce champ est facultatif et peut être utilisé pour spécifier un fichier de configuration personnalisé ou ajouter des métadonnées supplémentaires, telles que les détails de tâche, de définition de tâche, de cluster et d'instance de conteneur à l'événement de journal. Si spécifié, la syntaxe à utiliser est `"options":{"enable-ecs-log-metadata":"true|false","config-file-type:"s3|file","config-file-value":"arn:aws:s3:::amzn-s3-demo-bucket/fluent.conf|filepath"}`. Pour de plus amples informations, veuillez consulter Exemple de définition de tâche Amazon ECS : acheminer les journaux vers FireLens.

`type`

Type : String

Obligatoire : oui

Routeur de journal à utiliser. Les valeurs valides sont `fluentd` ou `fluentbit`.

Sécurité

Pour plus d'informations sur la sécurité des conteneurs, consultez les meilleures pratiques en matière de sécurité des conteneurs et des tâches Amazon ECS.

`credentialSpecs`

Type : tableau de chaînes

Obligatoire : non

Liste des informations contenues ARNs dans SSM ou Amazon S3 dans un fichier de spécifications d'identification (`CredSpec`) qui configure le conteneur pour l'authentification Active Directory. Nous vous recommandons d'utiliser ce paramètre plutôt que les `dockerSecurityOptions`. Le nombre maximum de ARNs est 1.

Il existe deux formats pour chaque ARN.

credentialspecdomainless:MyARN

Vous utilisez `credentialspecdomainless:MyARN` pour fournir un `CredSpec` avec une section supplémentaire pour un secret dans Secrets Manager. Vous fournissez les informations d'identification de connexion au domaine dans le secret.

Chaque tâche exécutée sur une instance de conteneur peut rejoindre différents domaines.

Vous pouvez utiliser ce format sans associer l'instance de conteneur à un domaine.

credentialspec:MyARN

Vous utilisez `credentialspec:MyARN` pour fournir un `CredSpec` pour un domaine unique.

Vous devez joindre l'instance de conteneur au domaine avant de démarrer toute tâche utilisant cette définition de tâche.

Dans les deux formats, remplacez `MyARN` par l'ARN dans SSM ou Amazon S3.

Le `credspec` doit fournir un ARN dans Secrets Manager pour un secret contenant le nom d'utilisateur, le mot de passe et le domaine auquel se connecter. Pour une meilleure sécurité, l'instance n'est pas jointe au domaine pour l'authentification sans domaine. Les autres applications de l'instance ne peuvent pas utiliser les informations d'identification sans domaine. Vous pouvez utiliser ce paramètre pour exécuter des tâches sur la même instance, même si les tâches doivent être jointes à différents domaines. Pour plus d'informations, consultez les sections Utilisation de g MSAs pour les conteneurs Windows et Utilisation de g MSAs pour les conteneurs Linux.

`user`

Type : chaîne

Obligatoire : non

Utilisateur à utiliser à l'intérieur du conteneur. Ce paramètre correspond `User` à la commande docker create-container et à l'option `--user` docker run.

> **Important**
>
> Lorsque vous exécutez des tâches en mode réseau `host`, n'exécutez pas de conteneurs à l'aide de l'utilisateur root (UID 0). Comme bonne pratique de sécurité, utilisez toujours un utilisateur non root.

Vous pouvez spécifier l'`user` à l'aide des formats suivants. Si vous spécifiez un UID ou GID, vous devez le spécifier en tant que nombre entier positif.

- `user`
- `user:group`
- `uid`
- `uid:gid`
- `user:gid`
- `uid:group`

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"user": "string"
```

Limites des ressources

`ulimits`

Type : tableau d'objets

Obligatoire : non

Une liste de valeurs `ulimit` à définir pour un conteneur. Cette valeur remplace le paramètre de quota de ressources par défaut pour le système d'exploitation. Ce paramètre correspond `Ulimits` à la commande docker create-container et à l'option `--ulimit` docker run.

Les tâches Amazon ECS hébergées sur Fargate utilisent les valeurs de limite définies par défaut par le système d'exploitation, à l'exception du paramètre de limite de ressources `nofile`. La

limite de ressources `nofile` restreint le nombre de fichiers ouverts qu'un conteneur peut utiliser. Sur Fargate, la limite flexible par défaut `nofile` est `65535` et la limite stricte est `65535`. Vous pouvez définir les valeurs des deux limites jusqu'à `1048576`. Pour de plus amples informations, veuillez consulter Limites des ressources de tâche.

Ce paramètre nécessite la version 1.18 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"ulimits": [
      {
        "name":
 "core"|"cpu"|"data"|"fsize"|"locks"|"memlock"|"msgqueue"|"nice"|"nofile"|"nproc"|"rss"|"rtp
        "softLimit": integer,
        "hardLimit": integer
      }
      ...
    ]
```

`name`

Type : String

Valeurs valides : `"core" | "cpu" | "data" | "fsize" | "locks" | "memlock" | "msgqueue" | "nice" | "nofile" | "nproc" | "rss" | "rtprio" | "rttime" | "sigpending" | "stack"`

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

Le `type` d'`ulimit`.

`hardLimit`

Type : entier

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

La limite stricte du type `ulimit`. La valeur peut être spécifiée en octets, en secondes ou en tant que nombre, selon le `typeulimit`.

`softLimit`

Type : entier

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

La limite flexible du type `ulimit`. La valeur peut être spécifiée en octets, en secondes ou en tant que nombre, selon le `typeulimit`.

Étiquettes Docker

`dockerLabels`

Type : mappage chaîne/chaîne

Obligatoire : non

Une key/value carte des étiquettes à ajouter au conteneur. Ce paramètre correspond `Labels` à la commande docker create-container et à l'option `--label` docker run.

Ce paramètre nécessite la version 1.18 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

```
"dockerLabels": {"string": "string"
      ...}
```

Autres paramètres de définition de conteneur

Les paramètres de définition de conteneur suivants peuvent être utilisés lors de l'enregistrement des définitions de tâche dans la console Amazon ECS à l'aide de l'option Configure via JSON (Configurer via JSON). Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.

Rubriques

- Paramètres Linux
- Dépendances du conteneur

- Temporisations de conteneurs
- Contrôles système
- Interactive
- Pseudo Terminal

Paramètres Linux

`linuxParameters`

Type : objet LinuxParameters

Obligatoire : non

Linux-des options spécifiques appliquées au conteneur, telles KernelCapabilitiesque.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"linuxParameters": {
      "capabilities": {
        "add": ["string", ...],
        "drop": ["string", ...]
        }
      }
```

`capabilities`

Type : objet KernelCapabilities

Obligatoire : non

Les Linux fonctionnalités du conteneur qui sont supprimées de la configuration par défaut fournie parDocker. Pour plus d'informations sur ces caractéristiques Linux, veuillez consulter la page consacrée aux caractéristiques(7) dans le manuel Linux (langue française non garantie).

`add`

Type : tableau de chaînes

Valeurs valides : `"SYS_PTRACE"`

Obligatoire : non

Caractéristiques Linux du conteneur à ajouter à la configuration par défaut fournie par Docker. Ce paramètre correspond `CapAdd` à la commande docker create-container et à l'option `--cap-add` docker run.

`drop`

Type : tableau de chaînes

Valeurs valides : `"ALL" | "AUDIT_CONTROL" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" | "NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW" | "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" | "SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT" | "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" | "SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"`

Obligatoire : non

Caractéristiques Linux du conteneur à supprimer de la configuration par défaut fournie par Docker. Ce paramètre correspond `CapDrop` à la commande docker create-container et à l'option `--cap-drop` docker run.

`devices`

Tous les périphériques hôtes à exposer au conteneur. Ce paramètre correspond `Devices` à la commande docker create-container et à l'option `--device` docker run.

> **Note**
>
> Le `devices` paramètre n'est pas pris en charge lorsque vous utilisez le type de lancement Fargate.

Type : tableau d'objets Périphérique

Obligatoire : non

`hostPath`

Chemin du périphérique dans l'instance de conteneur hôte.

Type : String

Obligatoire : oui

`containerPath`

Chemin auquel exposer l'appareil hôte à l'intérieur du conteneur.

Type : chaîne

Obligatoire : non

`permissions`

Autorisations explicites à fournir au conteneur pour le périphérique. Par défaut, le conteneur dispose des autorisations `read`, `write` et `mknod` sur l'appareil.

Type : tableau de chaînes

Valeurs valides : `read` | `write` | `mknod`

`initProcessEnabled`

Exécutez un processus `init` dans le conteneur afin de transmettre des signaux et de récolter les processus. Ce paramètre correspond à l'`--init`option docker run.

Ce paramètre nécessite la version 1.25 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

`maxSwap`

Cela n'est pas pris en charge pour les tâches exécutées sur Fargate.

Quantité totale de mémoire d'échange (en Mio) qu'un conteneur peut utiliser. Ce paramètre est traduit en `--memory-swap` option docker run où la valeur est la somme de la mémoire du conteneur plus la `maxSwap` valeur.

Si la valeur `0` est spécifiée pour `maxSwap`, le conteneur n'utilise pas l'échange. Les valeurs acceptées sont `0` ou n'importe quel nombre entier positif. Si le paramètre `maxSwap` n'est pas spécifié, le conteneur utilise la configuration d'échange pour l'instance de conteneur sur laquelle il s'exécute. Une valeur `maxSwap` doit être définie pour que le paramètre `swappiness` soit utilisé.

`sharedMemorySize`

Valeur correspondant à la taille (en Mio) du volume `/dev/shm`. Ce paramètre correspond à l'`--shm-size`option docker run.

> **Note**
>
> Si vous utilisez des tâches qui ont recours au type de lancement Fargate, le paramètre `sharedMemorySize` n'est pas pris en charge.

Type : entier

`tmpfs`

Chemin du conteneur, options de montage et taille maximale (en Mio) du montage tmpfs. Ce paramètre correspond à l'`--tmpfs`option docker run.

> **Note**
>
> Si vous utilisez des tâches qui ont recours au type de lancement Fargate, le paramètre `tmpfs` n'est pas pris en charge.

Type : tableau d'objets Tmpfs

Obligatoire : non

`containerPath`

Chemin de fichier absolu où sera monté le volume tmpfs.

Type : String

Obligatoire : oui

`mountOptions`

Liste des options de montage du volume tmpfs.

Type : tableau de chaînes

Obligatoire : non

Valeurs valides : `"defaults" | "ro" | "rw" | "suid" | "nosuid" | "dev" | "nodev" | "exec" | "noexec" | "sync" | "async" | "dirsync" | "remount" | "mand" | "nomand" | "atime" | "noatime" | "diratime" | "nodiratime" | "bind" | "rbind" | "unbindable" | "runbindable" | "private" | "rprivate" | "shared" | "rshared" | "slave" | "rslave" | "relatime" | "norelatime" | "strictatime" | "nostrictatime" | "mode" | "uid" | "gid" | "nr_inodes" | "nr_blocks" | "mpol"`

`size`

Taille maximale (en Mio) du volume tmpfs.

Type : entier

Obligatoire : oui

Dépendances du conteneur

`dependsOn`

Type : tableau d'objets ContainerDependency

Obligatoire : non

Dépendances définies pour le démarrage et l'arrêt de conteneurs. Un conteneur peut contenir plusieurs dépendances. Lorsqu'une dépendance est définie pour le démarrage des conteneurs, elle est inversée pour leur arrêt. Pour obtenir un exemple, consultez Dépendances du conteneur.

> **Note**
>
> Si un conteneur ne respecte pas une contrainte de dépendance ou s'écoule avant de respecter la contrainte, Amazon ECS ne fait pas progresser pas les conteneurs dépendants vers leur état suivant.

Ce paramètre nécessite que la tâche ou le service utilise une version de plate-forme `1.3.0` ou ultérieure (Linux) ou `1.0.0` (Windows).

```
"dependsOn": [
    {
        "containerName": "string",
```

```
        "condition": "string"
    }
]
```

`containerName`

Type : String

Obligatoire : oui

Nom de conteneur qui doit répondre à la condition spécifiée.

`condition`

Type : String

Obligatoire : oui

Condition de dépendance du conteneur. Voici les conditions disponibles et leur comportement :

- `START` - Cette condition émule le comportement de liens et de volumes aujourd'hui. La condition vérifie qu'un conteneur dépendant est démarré avant d'autoriser d'autres conteneurs à démarrer.
- `COMPLETE` - Cette condition vérifie qu'un conteneur dépendant exécute un cycle complet (sorties) avant d'autoriser d'autres conteneurs à démarrer. Cela peut être utile pour les conteneurs non essentiels qui exécutent un script, puis se ferment. Cette condition ne peut pas être définie sur un conteneur essentiel.
- `SUCCESS` - Cette condition est identique à `COMPLETE`, mais elle nécessite également que le conteneur se termine par un état `zero`. Cette condition ne peut pas être définie sur un conteneur essentiel.
- `HEALTHY` - Cette condition vérifie que la surveillance de l'état du conteneur dépendant réussit avant d'autoriser d'autres conteneurs à démarrer. Cela nécessite que le conteneur dépendant dispose de surveillances de l'état configurées dans la définition de la tâche. Cette condition est confirmée uniquement au démarrage de tâche.

Temporisations de conteneurs

`startTimeout`

Type : entier

Obligatoire : non

Exemples de valeur : `120`

Durée d'attente (en secondes) avant l'abandon de la résolution de dépendances pour un conteneur.

Par exemple, vous spécifiez deux conteneurs dans une définition de tâche, où le `containerA` dispose d'une dépendance au `containerB` avec un état `COMPLETE`, `SUCCESS` ou `HEALTHY`. Si une valeur `startTimeout` est spécifiée pour `containerB` et qu'il n'a pas atteint l'état souhaité dans ce délai, alors le `containerA` ne démarre pas.

> **Note**
>
> Si un conteneur ne respecte pas une contrainte de dépendance ou s'écoule avant de respecter la contrainte, Amazon ECS ne fait pas progresser pas les conteneurs dépendants vers leur état suivant.

Ce paramètre nécessite que la tâche ou le service utilise une version de plate-forme `1.3.0` ou ultérieure (Linux). La valeur maximale est de 120 secondes.

`stopTimeout`

Type : entier

Obligatoire : non

Exemples de valeur : `120`

Durée (en secondes) à attendre avant que le conteneur soit expressément tué s'il ne s'achève pas normalement tout seul.

Ce paramètre nécessite que la tâche ou le service utilise une version de plate-forme `1.3.0` ou ultérieure (Linux). Si le paramètre n'est pas spécifié, alors la valeur par défaut de 30 secondes est utilisée. La valeur maximale est de 120 secondes.

Contrôles système

`systemControls`

Type : objet SystemControl

Obligatoire : non

Liste des paramètres du noyau de l'espace de noms à définir dans le conteneur. Ce paramètre correspond `Sysctls` à la commande docker create-container et à l'option `--sysctl` docker run. Par exemple, vous pouvez configurer le paramètre `net.ipv4.tcp_keepalive_time` pour maintenir des connexions de plus longue durée.

Il n'est pas recommandé de spécifier des paramètres `systemControls` liés au réseau pour plusieurs conteneurs dans une seule tâche qui utilise également le mode réseau `awsvpc` ou `host`. En procédant ainsi, vous vous exposez aux inconvénients suivants :

- Si vous définissez `systemControls` une valeur pour un conteneur, elle s'applique à tous les conteneurs de la tâche. Si vous définissez différents paramètres `systemControls` pour plusieurs conteneurs dans une seule tâche, c'est le conteneur qui est démarré en dernier qui détermine le paramètre `systemControls` qui prend effet.

Si vous définissez un espace de noms des ressources IPC à utiliser pour les conteneurs de la tâche, les conditions suivantes s'appliquent aux contrôles de système. Pour de plus amples informations, veuillez consulter Mode IPC.

- Pour les tâches qui utilisent le mode IPC `host`, les `systemControls` liés à l'espace de noms IPC ne sont pas pris en charge.
- Pour les tâches qui utilisent le mode IPC `task`, les valeurs des `systemControls` liés à l'espace de noms IPC s'appliquent à tous les conteneurs au sein d'une tâche.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

> **Note**
>
> Ce paramètre est uniquement pris en charge pour les tâches hébergées sur AWS Fargate , si elles utilisent la version de plateforme `1.4.0` ou ultérieure (Linux). Cela n'est pas pris en charge par les conteneurs Windows sur Fargate.

```
"systemControls": [
    {
         "namespace":"string",
```

```
        "value":"string"
    }
]
```

`namespace`

Type : chaîne

Obligatoire : non

Le paramètre du noyau de l'espace de noms pour lequel définir un `value` nom.

Valeurs d'espace de noms IPC valides : `"kernel.msgmax" | "kernel.msgmnb" | "kernel.msgmni" | "kernel.sem" | "kernel.shmall" | "kernel.shmmax" | "kernel.shmmni" | "kernel.shm_rmid_forced"`, et `Sysctls` qui commencent par `"fs.mqueue.*"`

Valeurs d'espace de noms réseau valides : `Sysctls` qui commencent `"net.*"` par. Sur Fargate, seuls les espaces de `Sysctls` noms existant dans le conteneur sont acceptés.

Toutes ces valeurs sont prises en charge par Fargate.

`value`

Type : chaîne

Obligatoire : non

La valeur du paramètre de noyau de l'espace de noms spécifié dans`namespace`.

Interactive

`interactive`

Type : booléen

Obligatoire : non

Lorsque ce paramètre est `true`, vous pouvez déployer des applications conteneurisées qui nécessitent l'allocation d'une `stdin` ou d'un `tty`. Ce paramètre correspond `OpenStdin` à la commande docker create-container et à l'option `--interactive` docker run.

L'argument par défaut est `false`.

Pseudo Terminal

`pseudoTerminal`

Type : booléen

Obligatoire : non

Lorsque ce paramètre a la valeur `true`, un TTY est alloué. Ce paramètre correspond `Tty` à la commande docker create-container et à l'option `--tty` docker run.

L'argument par défaut est `false`.

Nom de l'accélérateur Elastic Inference

L'exigence de la ressource d'accélérateur Elastic Inference pour votre définition de tâche.

> **Note**
> Amazon Elastic Inference (EI) n'est plus disponible pour les clients.

Les paramètres suivants sont autorisés dans une définition de tâche :

`deviceName`

Type : String

Obligatoire : oui

Nom de l'appareil accélérateur d'inférence élastique pour l'instance. `deviceName` doit également être référencé dans une définition du conteneur. Consultez `Elastic Inference accelerator`.

`deviceType`

Type : String

Obligatoire : oui

L'accélérateur Elastic Inference à utiliser.

Configuration du proxy

`proxyConfiguration`

Type : objet ProxyConfiguration

Obligatoire : non

Détails de configuration pour le proxy App Mesh.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "string",
    "properties": [
        {
           "name": "string",
           "value": "string"
        }
    ]
}
```

`type`

Type : String

Valeur valeurs : `APPMESH`

Obligatoire : non

Type de proxy. La seule valeur prise en charge est `APPMESH`.

`containerName`

Type : String

Obligatoire : oui

Nom du conteneur qui fait office de proxy App Mesh.

`properties`

Type : tableau d'objets KeyValuePair

Obligatoire : non

L'ensemble de paramètres de configuration réseau pour fournir le plug-in Container Network Interface (CNI), spécifié sous la forme de paires clé-valeur.

- `IgnoredUID` – (Obligatoire) ID d'utilisateur (UID) du conteneur de proxy, tel que défini par le paramètre `user` dans une définition de conteneur. Ce port est utilisé pour garantir que le proxy ignore son propre trafic. Si vous avez spécifié `IgnoredGID`, ce champ doit être vide.
- `IgnoredGID` – (Obligatoire) ID de groupe (GID) du conteneur de proxy, tel que défini par le paramètre `user` dans une définition de conteneur. Ce port est utilisé pour garantir que le proxy ignore son propre trafic. Si vous avez spécifié `IgnoredUID`, ce champ doit être vide.
- `AppPorts` - (Obligatoire) Liste des ports utilisés par l'application. Le trafic réseau vers ces ports est transmis à `ProxyIngressPort` et `ProxyEgressPort`.
- `ProxyIngressPort` – (Obligatoire) Spécifie le port auquel le trafic entrant vers `AppPorts` est dirigé.
- `ProxyEgressPort` – (Obligatoire) Spécifie le port auquel le trafic sortant de `AppPorts` est dirigé.
- `EgressIgnoredPorts` – (Obligatoire) Le trafic sortant accédant à ces ports spécifiés est ignoré et n'est pas redirigé vers le `ProxyEgressPort`. Cela peut être une liste vide.
- `EgressIgnoredIPs` – (Obligatoire) Le trafic sortant accédant à ces adresses IP spécifiées est ignoré et n'est pas redirigé vers le `ProxyEgressPort`. Cela peut être une liste vide.

`name`

Type : chaîne

Obligatoire : non

Nom de la paire clé-valeur.

`value`

Type : chaîne

Obligatoire : non

Valeur de la paire clé-valeur.

Volumes

Lorsque vous enregistrez une définition de tâche, vous pouvez éventuellement spécifier une liste de volumes à transmettre au Docker démon sur une instance de conteneur, qui sera ensuite accessible aux autres conteneurs de la même instance de conteneur.

Les types de volumes de données qui peuvent être utilisés sont les suivants :

- Volumes Amazon EBS : fournit un stockage par blocs rentable, durable et performant pour les charges de travail conteneurisées gourmandes en données. Vous pouvez joindre 1 volume Amazon EBS par tâche Amazon ECS lors de l'exécution d'une tâche autonome, ou lors de la création ou de la mise à jour d'un service. Les volumes Amazon EBS sont pris en charge pour les tâches Linux hébergées sur Fargate. Pour de plus amples informations, veuillez consulter Utiliser des volumes Amazon EBS avec Amazon ECS.
- Volumes Amazon EFS : fournit un stockage de fichiers simple, évolutif et permanent à utiliser avec vos tâches Amazon ECS. Avec Amazon EFS, la capacité de stockage est élastique. Elle augmente et diminue automatiquement au fil de vos ajouts et suppressions de fichiers. Vos applications peuvent disposer de l'espace de stockage dont elles ont besoin, au moment où elles en ont besoin. Les volumes Amazon EFS sont pris en charge pour les tâches hébergées sur Fargate. Pour de plus amples informations, veuillez consulter Utiliser les volumes Amazon EFS avec Amazon ECS.
- FSx pour les volumes de serveurs de fichiers Windows : fournit des serveurs de fichiers Microsoft Windows entièrement gérés. Ces serveurs de fichiers sont basés sur un système de fichiers Windows. Lorsque vous utilisez FSx un serveur de fichiers Windows avec Amazon ECS, vous pouvez fournir à vos tâches Windows un stockage de fichiers persistant, distribué, partagé et statique. Pour de plus amples informations, veuillez consulter Utilisation FSx pour les volumes de serveurs de fichiers Windows avec Amazon ECS.

  Les conteneurs Windows sur Fargate ne prennent pas en charge cette option.
- Montages par liaison : fichier ou répertoire de la machine hôte monté dans un conteneur. Les volumes hôtes à montage par liaison sont pris en charge lors de l'exécution de tâches. Pour utiliser des volumes hôte de montage lié, spécifiez un `host` et éventuellement une valeur `sourcePath` dans votre définition de tâche.

Pour de plus amples informations, veuillez consulter Options de stockage pour les tâches Amazon ECS.

Les paramètres suivants sont autorisés dans une définition de conteneur.

`name`

Type : chaîne

Obligatoire : non

Nom du volume. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, tirets () et traits de soulignement (`-`) sont autorisés. _ Ce nom est référencé dans le `sourceVolume` paramètre de l'`mountPoints`objet de définition du conteneur.

`host`

Obligatoire : non

Le `host` paramètre est utilisé pour lier le cycle de vie du montage de liaison à l' EC2 instance Amazon hôte, plutôt qu'à la tâche et à l'endroit où il est stocké. Si le paramètre `host` est vide, le démon Docker attribue un chemin hôte au volume de données, mais la persistance des données après l'arrêt des conteneurs qui lui sont associés n'est pas garantie.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`.

> **Note**
>
> Le `sourcePath` paramètre n'est pris en charge que lors de l'utilisation de tâches hébergées sur EC2 des instances Amazon.

`sourcePath`

Type : chaîne

Obligatoire : non

Lorsque le `host` paramètre est utilisé, spécifiez a `sourcePath` pour déclarer le chemin sur l' EC2 instance Amazon hôte qui est présentée au conteneur. Si ce paramètre est vide, le démon Docker attribue un chemin hôte pour vous. Si le `host` paramètre contient un emplacement de `sourcePath` fichier, le volume de données est conservé à l'emplacement spécifié sur l' EC2 instance Amazon hôte jusqu'à ce que vous le supprimiez manuellement. Si la `sourcePath` valeur n'existe pas sur l' EC2 instance Amazon hôte, le démon Docker la crée. Si l'emplacement n'existe pas, le contenu du chemin source est exporté.

`configuredAtLaunch`

Type : booléen

Obligatoire : non

Spécifie si un volume est configurable au lancement. Lorsque ce paramètre est défini sur`true`, vous pouvez configurer le volume lors de l'exécution d'une tâche autonome ou lors de la création ou de la mise à jour d'un service. Lorsque ce paramètre est défini sur`true`, vous ne pourrez pas fournir d'autre configuration de volume dans la définition de tâche. Ce paramètre doit être défini sur `true` pour configurer un volume Amazon EBS à associer à une tâche. Le paramétrage `configuredAtLaunch` `true` et le report de la configuration du volume jusqu'à la phase de lancement vous permettent de créer des définitions de tâches qui ne sont pas limitées à un type de volume ou à des paramètres de volume spécifiques. Cela rend votre définition de tâche réutilisable dans différents environnements d'exécution. Pour plus d'informations, consultez Volumes Amazon EBS.

`dockerVolumeConfiguration`

Type : DockerVolumeConfigurationObjet

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Docker. Les volumes Docker ne sont pris en charge que lors de l'exécution de tâches sur des EC2 instances. Les conteneurs Windows ne prennent en charge que l'utilisation du `local` pilote. Pour utiliser des montages liés, spécifiez plutôt un paramètre `host`.

`scope`

Type : String

Valeurs valides : `task` | `shared`

Obligatoire : non

Portée du volume Docker, qui détermine son cycle de vie. Les volumes Docker destinés à un élément `task` sont automatiquement mis en service lorsque la tâche commence, et détruits lorsque la tâche s'arrête. Les volumes Docker définis comme `shared` ne sont pas supprimés lorsque la tâche s'arrête.

`autoprovision`

Type : booléen

Valeur par défaut : `false`

Obligatoire : non

Si cette valeur est `true`, le volume Docker est créé s'il n'existe pas déjà. Ce champ n'est utilisé que si `scope` c'est le cas`shared`. Si tel `scope` est le cas`task`, ce paramètre doit être omis.

`driver`

Type : chaîne

Obligatoire : non

Pilote de volume Docker à utiliser. La valeur du pilote doit correspondre au nom du pilote fourni par Docker car ce nom est utilisé pour le placement des tâches. Si le pilote a été installé à l'aide de la CLI du plugin Docker, utilisez-le `docker plugin ls` pour récupérer le nom du pilote depuis votre instance de conteneur. Si le pilote a été installé à l'aide d'une autre méthode, utilisez Docker plugin discovery pour récupérer le nom du pilote.

`driverOpts`

Type : chaîne

Obligatoire : non

Une carte des options spécifiques au pilote Docker à utiliser. Ce paramètre correspond `DriverOpts` à la section Créer un volume de Docker.

`labels`

Type : chaîne

Obligatoire : non

Métadonnées personnalisées à ajouter à votre volume Docker.

`efsVolumeConfiguration`

Type : objet EFSVolumede configuration

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Amazon EFS.

`fileSystemId`

Type : String

Obligatoire : oui

ID du système de fichiers Amazon EFS à utiliser.

`rootDirectory`

Type : chaîne

Obligatoire : non

Répertoire du système de fichiers Amazon EFS à monter en tant que répertoire racine à l'intérieur de l'hôte. Si ce paramètre est omis, la racine du volume Amazon EFS est utilisée. La spécification de `/` a le même effet que l'omission de ce paramètre.

> **Important**
>
> Si un point d'accès EFS est spécifié dans le`authorizationConfig`, le paramètre du répertoire racine doit être omis ou défini sur`/`, ce qui appliquera le chemin défini sur le point d'accès EFS.

`transitEncryption`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Indique si vous souhaitez activer ou non le chiffrement des données Amazon EFS en transit entre l'hôte Amazon ECS et le serveur Amazon EFS. Si l'autorisation Amazon EFS IAM est utilisée, le chiffrement de transit doit être activé. Si ce paramètre est omis, la valeur par défaut `DISABLED` est utilisée. Pour plus d'informations, consultez Chiffrement des données en transit dans le Guide de l'utilisateur Amazon Elastic File System.

`transitEncryptionPort`

Type : entier

Obligatoire : non

Port à utiliser lors de l'envoi de données chiffrées entre l'hôte Amazon ECS et le serveur Amazon EFS. Si vous ne spécifiez pas de port de chiffrement de transit, la tâche utilisera la stratégie de sélection de port utilisée par l'assistant de montage Amazon EFS. Pour plus d'informations, consultez Assistant de montage EFS dans le Guide de l'utilisateur Amazon Elastic File System User.

`authorizationConfig`

Type : objet EFSAuthorizationConfig

Obligatoire : non

Détails de configuration des autorisations pour le système de fichiers Amazon EFS.

`accessPointId`

Type : chaîne

Obligatoire : non

ID du point d'accès à utiliser. Si un point d'accès est spécifié, la valeur du répertoire racine `efsVolumeConfiguration` doit être omise ou définie sur`/`, ce qui appliquera le chemin défini sur le point d'accès EFS. Si un point d'accès est utilisé, le chiffrement de transit doit être activé dans `EFSVolumeConfiguration`. Pour plus d'informations, consultez Utilisation des points d'accès Amazon EFS dans le Guide de l'utilisateur Amazon Elastic File System.

`iam`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Spécifie s'il faut utiliser le rôle IAM de tâche Amazon ECS défini dans une définition de tâche lors du montage du système de fichiers Amazon EFS. Si cette option est activée, le chiffrement en transit doit être activé dans la configuration `EFSVolumeConfiguration`. Si ce paramètre est omis, la valeur par défaut `DISABLED` est utilisée. Pour plus d'informations, consultez Rôles IAM pour les tâches.

`FSxWindowsFileServerVolumeConfiguration`

Type : FSxWindowsFileServerVolumeConfigurationObjet

Obligatoire : oui

Ce paramètre est spécifié lorsque vous utilisez un système de fichiers Amazon FSx pour Windows File Server pour le stockage des tâches.

`fileSystemId`

Type : String

Obligatoire : oui

L' FSx identifiant du système de fichiers Windows File Server à utiliser.

`rootDirectory`

Type : String

Obligatoire : oui

Le répertoire du système FSx de fichiers Windows File Server à monter en tant que répertoire racine sur l'hôte.

`authorizationConfig`

`credentialsParameter`

Type : String

Obligatoire : oui

Options d'informations d'identification d'autorisation.

Options :

- Amazon Resource Name (ARN) d'un AWS Secrets Managersecret.
- ARN d'un AWS Systems Managerparamètre.

`domain`

Type : String

Obligatoire : oui

Nom de domaine complet hébergé par un annuaire AWS Directory Service for Microsoft Active Directory(AWS Managed Microsoft AD) ou un Active Directory auto-hébergé EC2 .

Balises

Lorsque vous enregistrez une définition de tâche, vous pouvez éventuellement spécifier des étiquettes de métadonnées qui sont appliquées à la définition de tâche. Les balises vous aident à classer et à organiser votre définition de tâche. Chaque balise est constituée d'une clé et d'une valeur facultative. Vous définissez ces deux éléments. Pour de plus amples informations, veuillez consulter Balisage des ressources Amazon ECS.

> **Important**
>
> N'ajoutez pas de données d'identification personnelle ou d'autres informations confidentielles ou sensibles dans les étiquettes. Les tags sont accessibles à de nombreux AWS services, y compris la facturation. Les étiquettes ne sont pas destinées à être utilisées pour des données privées ou sensibles.

Les paramètres suivants sont autorisés dans un objet balise.

`key`

Type : chaîne

Obligatoire : non

Partie d'une paire clé-valeur qui constitue une étiquette. Une clé est une étiquette générale qui fait office de catégorie pour les valeurs d'étiquette plus spécifiques.

`value`

Type : chaîne

Obligatoire : non

Partie facultative d'une paire clé-valeur qui constitue une étiquette. Une valeur agit comme un descripteur au sein d'une catégorie d'étiquette (clé).

Autres paramètres de définition de tâche

Les paramètres de définition de tâche suivants peuvent être utilisés lors de l'enregistrement des définitions de tâches dans la console Amazon ECS à l'aide de l'option Configure via JSON (Configurer via JSON). Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.

Rubriques

- Stockage éphémère
- Mode IPC
- Mode PID
- Injection de défauts

Stockage éphémère

`ephemeralStorage`

Type : objet EphemeralStorage

Obligatoire : non

Quantité de stockage éphémère (en Go) à allouer pour la tâche. Ce paramètre est utilisé pour étendre la quantité totale de stockage éphémère disponible, au-delà de la quantité par défaut, pour les tâches hébergées sur AWS Fargate. Pour de plus amples informations, veuillez consulter the section called “Montages liés”.

> **Note**
>
> Ce paramètre n'est pris en charge que sur la version de plate-forme `1.4.0` ou ultérieure (Linux) `1.0.0` ou ultérieure (Windows).

Mode IPC

`ipcMode`

Cela n'est pas pris en charge pour les tâches exécutées sur Fargate.

Type : chaîne

Obligatoire : non

Espace de noms de ressource IPC à utiliser pour les conteneurs de la tâche. Les valeurs valides sont `host`, `task` ou `none`. Si cette option `host` est spécifiée, tous les conteneurs inclus dans les tâches qui ont spécifié le mode `host` IPC sur la même instance de conteneur partagent les mêmes ressources IPC avec l'instance Amazon EC2 hôte. Si `task` est spécifié, tous les conteneurs de la tâche spécifiée partagent les mêmes ressources IPC. Si `none` est spécifié, les ressources IPC des conteneurs d'une tâche sont privés et ne partagent rien avec les autres conteneurs d'une tâche ou d'une instance de conteneur. Si aucune valeur n'est spécifiée, le partage de l'espace de noms de ressource IPC dépend du paramètre du démon Docker sur l'instance de conteneur.

Si le mode IPC `host` est utilisé, il existe un risque accru d'exposition à un espace de noms IPC indésirable.

Si vous définissez les paramètres du noyau placés dans un espace de noms à l'aide de `systemControls` pour les conteneurs de la tâche, les éléments suivants s'appliquent à votre espace de noms de ressource IPC.

- Pour les tâches qui utilisent le mode IPC `host`, les `systemControls` liés à l'espace de noms IPC ne sont pas pris en charge.
- Pour les tâches qui utilisent le mode IPC `task`, les `systemControls` liés à l'espace de noms IPC s'appliquent à tous les conteneurs au sein d'une tâche.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le type de lancement Fargate.

Mode PID

`pidMode`

Type : String

Valeurs valides : `host` | `task`

Obligatoire : non

Espace de noms de processus à utiliser pour les conteneurs de la tâche. Les valeurs valides sont `host` ou `task`. Sur les conteneurs Linux, la seule valeur valide est`task`. Par exemple, la surveillance des sidecars peut avoir besoin de `pidMode` pour accéder à des informations sur d'autres conteneurs exécutés dans le cadre de la même tâche.

Si `task` est spécifié, tous les conteneurs de la tâche spécifiée partagent le même espace de noms de processus.

Si aucune valeur n'est spécifiée, la valeur par défaut est un espace de noms privé pour chaque conteneur.

> **Note**
>
> Ce paramètre est uniquement pris en charge pour les tâches hébergées sur AWS Fargate , si elles utilisent la version de plateforme `1.4.0` ou ultérieure (Linux). Cela n'est pas pris en charge par les conteneurs Windows sur Fargate.

Injection de défauts

`enableFaultInjection`

Type : booléen

Valeurs valides : `true` | `false`

Obligatoire : non

Si ce paramètre est défini sur`true`, dans la charge utile d'une tâche, Amazon ECS et Fargate acceptent les demandes d'injection d'erreurs provenant des conteneurs de la tâche. Par défaut, ce paramètre est défini sur `false`.

Paramètres de définition des tâches Amazon ECS pour le type de EC2 lancement

Les définitions de tâches sont divisées en plusieurs parties : la famille de tâches, le rôle de tâche AWS Identity and Access Management (IAM), le mode réseau, les définitions des conteneurs, les

volumes, les contraintes de placement des tâches et les types de lancement. Les définitions de famille et de conteneur sont requises dans une définition de tâche. En revanche, le rôle de tâche, le mode réseau, les volumes, les contraintes de placement des tâches et le type de lancement sont facultatifs.

Vous pouvez utiliser ces paramètres dans un fichier JSON pour configurer votre définition de tâche.

Vous trouverez ci-dessous des descriptions plus détaillées de chaque paramètre de définition de tâche pour le type de EC2 lancement.

Famille

`family`

Type : String

Obligatoire : oui

Lorsque vous enregistrez une définition de tâche, vous lui attribuez une famille. Cela équivaut, pour plusieurs versions de définition de tâche, à lui attribuer un nom spécifié avec un numéro de révision. La première définition de tâche enregistrée dans une famille donnée reçoit le numéro de révision 1. Toute définition de tâche enregistrée après celle-ci reçoit un numéro de révision ultérieur dans l'ordre séquentiel.

Types de lancement

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier un type de lancement par rapport auquel Amazon ECS doit valider la définition de tâche. Une exception client est renvoyée si la définition de tâche n'est pas conforme aux compatibilités spécifiées. Pour de plus amples informations, veuillez consulter Types de lancement Amazon ECS.

Le paramètre suivant est autorisé dans une définition de tâche.

`requiresCompatibilities`

Type : tableau de chaînes

Obligatoire : non

Valeurs valides : `EC2`

Les types de lancement pour lesquels la définition de tâche doit être validée. Cela permet de lancer une vérification afin de s'assurer que tous les paramètres utilisés dans la définition de la tâche correspondent aux exigences du type de lancement.

Rôle de tâche

`taskRoleArn`

Type : chaîne

Obligatoire : non

Lorsque vous enregistrez une définition de tâche, vous pouvez attribuer un rôle de tâche à un rôle IAM qui permet aux conteneurs concernés par l'autorisation de tâche d'appeler AWS APIs les conteneurs spécifiés dans les politiques associées en votre nom. Pour de plus amples informations, veuillez consulter Rôle IAM de la tâche Amazon ECS.

Lorsque vous lancez l'AMI Windows Server optimisée pour Amazon ECS, les rôles IAM pour les tâches sous Windows nécessitent que l'option `-EnableTaskIAMRole` soit définie. Vos conteneurs doivent également exécuter certains codes de configuration afin d'utiliser la fonctionnalité. Pour de plus amples informations, veuillez consulter Configuration supplémentaire de l'instance Amazon EC2 Windows.

Rôle d'exécution de tâche

`executionRoleArn`

Type : String

Obligatoire : Conditionnelle

Nom de ressource Amazon (ARN) du rôle d'exécution des tâches qui autorise l'agent de conteneur Amazon ECS à effectuer des appels d' AWS API en votre nom.

> **Note**
>
> Le rôle IAM d'exécution de la tâche est requis en fonction des besoins de votre tâche. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

Mode réseau

`networkMode`

Type : chaîne

Obligatoire : non

Mode réseau Docker à utiliser pour les conteneurs de la tâche. Pour les tâches Amazon ECS hébergées sur des instances Amazon EC2 Linux, les valeurs valides sont `none`,`bridge`,`awsvpc`, et`host`. Si aucun mode réseau n'est spécifié, le mode réseau par défaut est`bridge`. Pour les tâches Amazon ECS hébergées sur des instances Amazon EC2 Windows, les valeurs valides sont `default` et`awsvpc`. Si aucun mode réseau n'est spécifié, c'est le mode `default` réseau qui est utilisé.

Si le mode réseau est défini sur `none`, les conteneurs de la tâche ne disposent d'aucune connectivité externe et les mappages de ports ne peuvent pas être spécifiés dans la définition du conteneur.

Si le mode réseau est activé`bridge`, la tâche utilise le réseau virtuel intégré de Docker sous Linux, qui s'exécute dans chaque EC2 instance Amazon hébergeant la tâche. Le réseau virtuel intégré sous Linux utilise le pilote réseau Docker `bridge`.

Si le mode réseau est activé`host`, la tâche utilise le réseau de l'hôte qui contourne le réseau virtuel intégré de Docker en mappant les ports du conteneur directement à l'ENI de l' EC2 instance Amazon qui héberge la tâche. Les mappages de ports dynamiques ne peuvent pas être utilisés dans ce mode réseau. Dans une définition de tâche qui utilise ce mode, un conteneur doit spécifier un numéro `hostPort` spécifique. Un numéro de port sur un hôte ne peut pas être utilisé par plusieurs tâches. Par conséquent, vous ne pouvez pas exécuter plusieurs tâches ayant la même définition de tâche sur une seule EC2 instance Amazon.

> **Important**
>
> Lorsque vous exécutez des tâches en mode réseau `host`, vous n'exécutez pas de conteneurs à l'aide de l'utilisateur root (UID 0) pour une meilleure sécurité. Comme bonne pratique de sécurité, utilisez toujours un utilisateur non root.

Si le mode réseau est `awsvpc`, une interface réseau Elastic est attribuée à la tâche, et vous devez spécifier une configuration `NetworkConfiguration` lorsque vous créez un service

ou exécutez une tâche avec la définition de tâche. Pour de plus amples informations, veuillez consulter Options de mise en réseau des tâches Amazon ECS pour le type de EC2 lancement.

Si le mode réseau est activé`default`, la tâche utilise le réseau virtuel intégré de Docker sous Windows, qui s'exécute dans chaque EC2 instance Amazon hébergeant la tâche. Le réseau virtuel intégré sous Windows utilise le pilote réseau Docker `nat`.

Les modes `awsvpc` réseau `host` et offrent les meilleures performances réseau pour les conteneurs, car ils utilisent la pile EC2 réseau Amazon. Avec les modes réseau `host` et `awsvpc`, les ports de conteneur exposés sont mappés directement au port hôte correspondant (pour le mode réseau hôte `host`) ou au port de l'interface réseau Elastic (pour le mode réseau `awsvpc`). Pour cette raison, vous ne pouvez pas utiliser de mappages de ports hôtes dynamiques.

Le mode réseau autorisé dépend du système d'exploitation de l' EC2 instance sous-jacente. Sous Linux, n'importe quel mode réseau peut être utilisé. S'il s'agit de Windows, les modes `default` et `awsvpc` peuvent être utilisés.

Plateforme d'exécution

`operatingSystemFamily`

Type : String

Obligatoire : Conditionnelle

Par défaut : LINUX

Lorsque vous enregistrez une définition de tâche, vous spécifiez la famille du système d'exploitation.

Les valeurs valides sont `LINUXWINDOWS_SERVER_2025_FULL`,`WINDOWS_SERVER_2025_CORE`,`WINDOWS_SERVER_2022_CORE` et `WINDOWS_SERVER_2019_COREWINDOWS_SERVER_2016_FULL`,`WINDOWS_SERVER_2004_CORE`, et`WINDOWS_SERVER_20H2_CORE`.

Toutes les définitions de tâches qui sont utilisées dans un service doivent avoir la même valeur pour ce paramètre.

Lorsqu'une définition de tâche fait partie d'un service, cette valeur doit correspondre à la valeur `platformFamily` du service.

`cpuArchitecture`

Type : String

Obligatoire : Conditionnelle

Par défaut : X86_64

Lorsque vous enregistrez une définition de tâche, vous spécifiez l'architecture du processeur. Les valeurs valides sont `X86_64` et `ARM64`.

Toutes les définitions de tâches qui sont utilisées dans un service doivent avoir la même valeur pour ce paramètre.

Lorsque vous avez des tâches Linux, vous pouvez définir la valeur sur`ARM64`. Pour de plus amples informations, veuillez consulter the section called “Définitions de tâches pour les charges de travail ARM 64 bits”.

Taille de la tâche

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier la quantité totale d'UC et de mémoire utilisée pour cette tâche. Ces valeurs sont distinctes des valeurs `cpu` et `memory` au niveau de la définition de conteneur. Pour les tâches hébergées sur des EC2 instances Amazon, ces champs sont facultatifs.

> **Note**
>
> Les paramètres d'UC et de mémoire de niveau tâche sont ignorés pour les conteneurs Windows. Nous vous recommandons de spécifier des ressources de niveau conteneur pour les conteneurs Windows.

`cpu`

Type : String

Obligatoire : Conditionnelle

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

Limite stricte du nombre d'unités UC à présenter pour la tâche. Vous pouvez spécifier les valeurs du processeur dans le fichier JSON sous forme de chaîne en unités de processeur ou en mode virtuel CPUs (vCPUs). Par exemple, vous pouvez spécifier une valeur de processeur soit `1024` en unités de processeur, soit `1 vCPU` en CPUs v. Lorsque la définition de tâche est enregistrée, une valeur vCPU est convertie en un entier indiquant les unités du processeur.

Ce champ est facultatif. Si votre cluster n'a pas d'instances de conteneur enregistrées avec les unités UC demandées disponibles, la tâche échoue. Les valeurs prises en charge sont comprises entre `0.125` `192` v CPUs et CPUs v.

`memory`

Type : String

Obligatoire : Conditionnelle

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

La limite stricte de mémoire à présenter à la tâche. Vous pouvez spécifier des valeurs de mémoire dans la définition de tâche sous forme de chaîne en mégaoctets (MiB) ou en gigaoctets (Go). Par exemple, vous pouvez spécifier une valeur de mémoire `3072` en MiB ou `3 GB` en Go. Lorsque la définition de tâche est enregistrée, la valeur en Go est convertie en un nombre entier indiquant la quantité de Mio.

Ce champ est facultatif et n'importe quelle valeur peut être utilisée. Si une valeur de mémoire au niveau de la tâche est spécifiée, la valeur de mémoire au niveau du conteneur est facultative. Si votre cluster n'a pas d'instances de conteneur enregistrées avec la mémoire demandée disponible, la tâche échoue. Vous pouvez optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier. Pour de plus amples informations, veuillez consulter Réservation de mémoire d'instance de conteneur Amazon ECS Linux.

Définitions de conteneur

Lorsque vous enregistrez une définition de tâche, vous devez spécifier une liste de définitions de conteneur qui sont transmises au démon Docker sur une instance de conteneur. Les paramètres suivants sont autorisés dans une définition de conteneur.

Rubriques

- Paramètres de définition de conteneur standards
- Paramètres de définition de conteneur avancés
- Autres paramètres de définition de conteneur

Paramètres de définition de conteneur standards

Les paramètres de définition de tâche suivants sont obligatoires ou utilisés dans la plupart des définitions de conteneur.

Rubriques

- Nom
- Image
- Mémoire
- Mappages de port
- Informations d'identification du référentiel privé

Nom

`name`

Type : String

Obligatoire : oui

Le nom d'un conteneur. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres, des tirets ou des traits de soulignement. Si vous associez plusieurs conteneurs dans une définition de tâche, vous pouvez spécifier l'option `name` d'un conteneur dans l'option `links` d'un autre conteneur. Cela permet de connecter les conteneurs.

Image

`image`

Type : String

Obligatoire : oui

Image utilisée pour démarrer un conteneur. Cette chaîne est transmise directement au démon Docker. Par défaut, les images dans le registre Docker Hub sont disponibles. Vous pouvez également spécifier d'autres référentiels avec soit *`repository-url`*`/`*`image`*`:`*`tag`*, soit *`repository-url`*`/`*`image`*`@`*`digest`*. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres, des tirets, des traits de soulignement, deux points, des points, des barres obliques et des signes dièse. Ce paramètre correspond `Image` à la commande docker create-container et au paramètre `IMAGE` de la commande docker run.

- Lorsqu'une nouvelle tâche démarre, l'agent de conteneur Amazon ECS extrait la version la plus récente de l'image et de l'étiquette spécifiées afin que le conteneur puisse les utiliser. Notez cependant que les mises à jour ultérieures apportées à une image de référentiel ne sont pas répercutées sur les tâches déjà en cours d'exécution.
- Lorsque vous ne spécifiez pas de balise ou de résumé dans le chemin de l'image dans la définition de tâche, l'agent de conteneur Amazon ECS extrait la dernière version de l'image spécifiée.
- Notez cependant que les mises à jour ultérieures apportées à une image de référentiel ne sont pas répercutées sur les tâches déjà en cours d'exécution.
- Les images des registres privés sont prises en charge. Pour de plus amples informations, veuillez consulter Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.
- Les images des référentiels Amazon ECR peuvent être spécifiées en utilisant soit la convention de dénomination complète `registry/repository:tag` ou `registry/repository@digest` (par exemple, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`* ou *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`@`*`sha256:94afd1f2e64d908bc90dbca0035a5b567EXAMPLE`*).
- Les images dans les référentiels officiels sur Docker Hub utilisent un nom unique (par exemple, `ubuntu` ou `mongo`).
- Les images dans les autres référentiels sur Docker Hub sont qualifiées par un nom d'organisation (par exemple, `amazon/amazon-ecs-agent`).

- Les images dans les autres référentiels en ligne sont qualifiées par un nom de domaine (par exemple, `quay.io/assemblyline/ubuntu`).

`versionConsistency`

Type : String

Valeurs valides : `enabled`|`disabled`

Obligatoire : non

Spécifie si Amazon ECS transformera la balise d'image de conteneur fournie dans la définition du conteneur en un condensé d'image. Par défaut, ce comportement est`enabled`. Si vous définissez la valeur d'un conteneur comme suit`disabled`, Amazon ECS ne transformera pas la balise d'image du conteneur en résumé et utilisera l'URI d'image d'origine spécifiée dans la définition du conteneur pour le déploiement. Pour plus d'informations sur la résolution de l'image du conteneur, consultezRésolution de l'image du conteneur.

Mémoire

`memory`

Type : entier

Obligatoire : non

La quantité de mémoire (en Mio) à présenter au conteneur. Si votre conteneur tente de dépasser la mémoire spécifiée ici, il sera désactivé. La quantité totale de mémoire réservée pour tous les conteneurs au sein d'une tâche doit être inférieure à la valeur `memory` de la tâche, si cette valeur est spécifiée. Ce paramètre correspond `Memory` à la commande docker create-container et à l'option `--memory` docker run.

Vous devez spécifier une valeur de mémoire au niveau de la tâche ou au niveau du conteneur. Si vous spécifiez à la fois une valeur de `memory` au niveau du conteneur et une valeur `memoryReservation`, la valeur de `memory` doit être supérieure à celle de `memoryReservation`. Si vous spécifiez `memoryReservation`, cette valeur est soustraite des ressources mémoire disponibles pour l'instance de conteneur sur laquelle le conteneur est placé. Sinon, c'est la valeur `memory` qui est utilisée.

Le démon Docker 20.10.0 ou ultérieur réserve un minimum de 6 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 6 Mio de mémoire pour vos conteneurs.

Le démon Docker 19.03.13-ce ou antérieur réserve un minimum de 4 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 4 Mio de mémoire pour vos conteneurs.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire d'instance de conteneur Amazon ECS Linux.

`memoryReservation`

Type : entier

Obligatoire : non

La limite flexible (en Mio) de mémoire à réserver pour le conteneur. En cas de contention de la mémoire système, Docker tente de garder la mémoire du conteneur en deçà de cette limite flexible. Toutefois, votre conteneur peut utiliser davantage de mémoire en cas de besoin. Le conteneur peut consommer jusqu'à la limite stricte spécifiée avec le paramètre `memory` (le cas échéant), ou la totalité de la mémoire disponible sur l'instance de conteneur, selon la première valeur atteinte. Ce paramètre correspond `MemoryReservation` à la commande docker create-container et à l'option `--memory-reservation` docker run.

Si aucune valeur de mémoire au niveau de la tâche n'est spécifiée, vous devez indiquer un nombre entier différent de zéro pour `memory` ou `memoryReservation` dans une définition de conteneur. Si vous spécifiez les deux, `memory` doit être supérieur à `memoryReservation`. Si vous spécifiez `memoryReservation`, cette valeur est soustraite des ressources mémoire disponibles pour l'instance de conteneur sur laquelle le conteneur est placé. Sinon, c'est la valeur `memory` qui est utilisée.

Par exemple, supposons que votre conteneur utilise normalement 128 Mio de mémoire, mais qu'il lui arrive d'utiliser jusqu'à 256 Mio de mémoire pendant de courtes périodes. Vous pouvez définir une `memoryReservation` de 128 Mio et une limite stricte `memory` de 300 Mio. Cette configuration permet au conteneur de ne réserver que 128 Mio de mémoire à partir des ressources restantes sur l'instance de conteneur. En même temps, cette configuration permet également au conteneur d'utiliser davantage de ressources mémoire lorsque cela est nécessaire.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

Le démon Docker 20.10.0 ou ultérieur réserve un minimum de 6 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 6 Mio de mémoire pour vos conteneurs.

Le démon Docker 19.03.13-ce ou antérieur réserve un minimum de 4 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 4 Mio de mémoire pour vos conteneurs.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire d'instance de conteneur Amazon ECS Linux.

Mappages de port

`portMappings`

Type : tableau d'objets

Obligatoire : non

Les mappages de ports exposent les ports réseau de votre conteneur au monde extérieur, ce qui permet aux clients d'accéder à votre application. Il est également utilisé pour la communication entre conteneurs dans le cadre d'une même tâche.

Pour les définitions de tâche qui utilisent le mode réseau `awsvpc`, spécifiez uniquement `containerPort`. Le `hostPort` est toujours ignoré et le port du conteneur est automatiquement mappé à un port aléatoire à numéro élevé sur l'hôte.

Les mappages de ports sur Windows utilisent l'adresse de passerelle `NetNAT` plutôt que `localhost`. Il n'existe aucune boucle pour les mappages de ports sous Windows, donc vous ne pouvez pas accéder au port mappé d'un conteneur à partir de l'hôte lui-même.

La plupart des champs de ce paramètre (y compris`containerPort`,`hostPort`,`protocol`) sont mappés `PortBindings` dans la commande docker create-container et dans l'option `--`

`publish` docker run. Si le mode réseau d'une définition de tâche est défini sur `host`, les ports hôtes doivent être indéfinis ou correspondre au port du conteneur dans le mappage de port.

> **Note**
>
> Une fois qu'une tâche passe à l'état `RUNNING`, les affectations manuelles et automatiques de ports de conteneur et d'hôte sont visibles aux emplacements suivants :
>
> - Console : la section Network Bindings (Liaisons réseau) de la description d'un conteneur pour une tâche sélectionnée.
> - AWS CLI : la section `networkBindings` de la sortie de la commande describe-tasks.
> - API : la réponse `DescribeTasks`.
> - Métadonnées : point de terminaison des métadonnées de la tâche.

`appProtocol`

Type : chaîne

Obligatoire : non

Protocole d'application utilisé pour le mappage de port. Ce paramètre s'applique uniquement à Service Connect. Nous vous conseillons de définir ce paramètre de manière cohérente avec le protocole que votre application utilise. Si vous définissez ce paramètre, Amazon ECS ajoute une gestion de connexion spécifique au protocole au proxy Service Connect. Si vous définissez ce paramètre, Amazon ECS ajoute une télémétrie spécifique au protocole dans la console Amazon ECS et. CloudWatch

Si vous ne définissez aucune valeur pour ce paramètre, le protocole TCP est utilisé. Toutefois, Amazon ECS n'ajoute pas de télémétrie spécifique au protocole pour TCP.

Pour de plus amples informations, veuillez consulter the section called “ Service Connect”.

Valeurs de protocole valides : `"http" | "http2" | "grpc"`

`containerPort`

Type : entier

Obligatoire : oui, lorsque des objets `portMappings` sont utilisés

Le numéro de port sur le conteneur qui est lié au port hôte spécifié par l'utilisateur ou affecté automatiquement.

Pour les tâches utilisant le mode `awsvpc` réseau, vous `containerPort` devez spécifier les ports exposés.

Supposons que vous utilisiez des conteneurs dans une tâche avec le type de EC2 lancement et que vous spécifiez un port de conteneur et non un port hôte. Ensuite, votre conteneur reçoit automatiquement un port hôte dans la plage de ports éphémères. Pour de plus amples informations, veuillez consulter `hostPort`. Les mappages de ports qui sont automatiquement affectés de cette façon ne sont pas comptabilisés dans le quota des 100 ports réservés d'une instance de conteneur.

`containerPortRange`

Type : chaîne

Obligatoire : non

Plage de numéros de port du conteneur liée à la plage de ports hôtes mappés dynamiquement.

Vous ne pouvez définir ce paramètre qu'à l'aide de l'API `register-task-definition`. L'option est disponible dans le paramètre `portMappings`. Pour plus d'informations, consultez register-task-definition dans la référence AWS Command Line Interface .

Les règles suivantes s'appliquent lorsque vous spécifiez une `containerPortRange` :

- Vous devez utiliser le mode réseau `bridge` ou le mode réseau `awsvpc`.
- Ce paramètre est disponible pour les systèmes d'exploitation Windows et Linux.
- L'instance de conteneur doit au moins disposer de la version 1.67.0 de l'agent de conteneur et au moins de la version 1.67.0-1 du package `ecs-init`.
- Vous pouvez spécifier un maximum de 100 plages de ports pour chaque conteneur.
- Vous ne spécifiez pas de `hostPortRange`. La valeur de `hostPortRange` est définie comme suit :
  - Pour les conteneurs d'une tâche en mode réseau `awsvpc`, le `hostPort` est défini sur la même valeur que le `containerPort`. Il s'agit d'une stratégie de mappage statique.
  - Pour les conteneurs d'une tâche en mode réseau `bridge`, l'agent Amazon ECS trouve les ports hôtes ouverts dans la plage éphémère par défaut et les transmet à Docker pour les lier aux ports du conteneur.

- Les valeurs valides de `containerPortRange` sont comprises entre 1 et 65535.
- Un port peut uniquement être inclus dans un mappage de port pour chaque conteneur.
- Vous ne pouvez pas spécifier de plages de ports qui se chevauchent.
- Le premier port de la plage doit être inférieur au dernier port de la plage.
- Docker vous recommande de désactiver le proxy Docker dans le fichier de configuration du démon Docker en présence d'un grand nombre de ports.

  Pour plus d'informations, consultez le numéro #11185 sur GitHub.

  Pour plus d'informations sur la désactivation du proxy Docker dans le fichier de configuration du démon Docker, veuillez consulter Démon Docker dans le Guide du développeur Amazon ECS (langue française non garantie).

Vous pouvez appeler DescribeTasks pour voir la `hostPortRange` correspondant aux ports hôtes liés aux ports de conteneur.

Les plages de ports ne sont pas incluses dans les événements de tâches Amazon ECS, qui sont envoyés à EventBridge. Pour de plus amples informations, veuillez consulter the section called “Automatisez les réponses aux erreurs Amazon ECS à l'aide de EventBridge”.

`hostPortRange`

Type : chaîne

Obligatoire : non

Plage de numéros de port sur l'hôte utilisée avec la liaison réseau. Elle est attribuée par Docker et délivrée par l'agent Amazon ECS.

`hostPort`

Type : entier

Obligatoire : non

Le numéro de port sur l'instance de conteneur à réserver pour votre conteneur.

Vous pouvez spécifier un port hôte non réservé pour le mappage de ports de votre conteneur. C'est ce que l'on appelle le mappage statique des ports hôtes. Vous pouvez également omettre le `hostPort` (ou le définir sur `0`) lorsque vous spécifiez un `containerPort`. Votre conteneur reçoit automatiquement un port dans la plage de ports éphémères pour le système

d'exploitation et la version Docker de votre instance de conteneur. C'est ce que l'on appelle le mappage dynamique des ports hôtes.

La plage de ports éphémères par défaut pour Docker 1.6.0 et versions ultérieures est répertoriée dans l'instance sous `/proc/sys/net/ipv4/ip_local_port_range`. Si ce paramètre de noyau n'est pas disponible, la plage de ports éphémères par défaut de `49153-65535` est utilisée. N'essayez pas de spécifier un port d'hôte dans la plage de ports éphémères. Cela est dû au fait qu'ils sont réservés à une affectation automatique. En général, les ports inférieurs à `32768` ne sont pas compris dans la plage de ports éphémères.

Les ports réservés par défaut sont le port `22` pour SSH, les ports Docker `2375` et `2376`, et les ports d'agent de conteneur Amazon ECS `51678-51680`. Tout port hôte ayant été spécifié précédemment par l'utilisateur pour une tâche en cours d'exécution est également réservé pendant l'exécution de la tâche. Après l'arrêt d'une tâche, le port hôte est libéré. Les ports actuellement réservés sont affichés dans `remainingResources` la describe-container-instances sortie. Une instance de conteneur peut avoir jusqu'à 100 ports réservés à la fois, y compris les ports réservés par défaut. Les ports affectés automatiquement ne sont pas pris en compte dans le quota des 100 ports réservés.

`name`

Type : String

Obligatoire : Non, obligatoire pour que Service Connect et VPC Lattice soient configurés dans un service

Nom utilisé pour le mappage de port. Ce paramètre s'applique uniquement à Service Connect et à VPC Lattice. Ce paramètre est le nom que vous utilisez dans la configuration Service Connect et VPC Lattice d'un service.

Pour de plus amples informations, veuillez consulter Utilisez Service Connect pour connecter les services Amazon ECS avec des noms abrégés.

Dans l'exemple suivant, les deux champs obligatoires pour Service Connect et VPC Lattice sont utilisés.

```
"portMappings": [
    {
        "name": string,
        "containerPort": integer
```

```
    }
]
```

`protocol`

Type : chaîne

Obligatoire : non

Le protocole utilisé pour le mappage de port. Les valeurs valides sont `tcp` et `udp`. L'argument par défaut est `tcp`.

> **Important**
>
> Seul `tcp` est pris en charge pour Service Connect. N'oubliez pas que `tcp` est implicite si ce champ n'est pas défini.

> **Important**
>
> La prise en charge du protocole UDP n'est disponible que sur les instances de conteneur qui ont été lancées avec la version 1.2.0 ou ultérieure de l'agent de conteneur Amazon ECS (comme l'AMI `amzn-ami-2015.03.c-amazon-ecs-optimized`), ou avec des agents de conteneur qui ont été mis à jour vers la version 1.3.0 ou ultérieure. Pour mettre à jour votre agent de conteneur avec la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Si vous spécifiez un port hôte, utilisez la syntaxe suivante.

```
"portMappings": [
    {
        "containerPort": integer,
        "hostPort": integer
    }
    ...
]
```

Si vous souhaitez utiliser un port hôte affecté automatiquement, utilisez la syntaxe suivante.

```
"portMappings": [
    {
        "containerPort": integer
    }
    ...
]
```

Informations d'identification du référentiel privé

`repositoryCredentials`

Type : objet RepositoryCredentials

Obligatoire : non

Informations d'identification du référentiel pour l'authentification de registre privé.

Pour de plus amples informations, veuillez consulter Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.

`credentialsParameter`

Type : String

Obligatoire : oui, lorsque des objets `repositoryCredentials` sont utilisés

L'Amazon Resource Name (ARN) du secret contenant les informations d'identification du référentiel privé.

Pour de plus amples informations, veuillez consulter Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.

> **Note**
>
> Lorsque vous utilisez l'API Amazon ECS AWS CLI AWS SDKs, ou si le secret existe dans la même région que la tâche que vous lancez, vous pouvez utiliser l'ARN complet ou le nom du secret. Lorsque vous utilisez le AWS Management Console, vous devez spécifier l'ARN complet du secret.

Voici un extrait de code d'une définition de tâche indiquant les paramètres requis :

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
        }
    }
]
```

Paramètres de définition de conteneur avancés

Les paramètres de définition de conteneur avancés suivants fournissent des fonctionnalités étendues à la commande docker run utilisée pour lancer des conteneurs sur vos instances de conteneur Amazon ECS.

Rubriques

- Politique de redémarrage
- Surveillance de l'état
- Environnement
- Paramètres réseau
- Stockage et journalisation
- Sécurité
- Limites des ressources
- Étiquettes Docker

Politique de redémarrage

`restartPolicy`

La politique de redémarrage du conteneur et les paramètres de configuration associés. Lorsque vous configurez une politique de redémarrage pour un conteneur, Amazon ECS peut redémarrer le conteneur sans avoir à remplacer la tâche. Pour de plus amples informations, veuillez consulter Redémarrer des conteneurs individuels dans des tâches Amazon ECS avec des politiques de redémarrage de conteneurs.

`enabled`

Type : booléen

Obligatoire : oui

Spécifie si une politique de redémarrage est activée pour le conteneur.

`ignoredExitCodes`

Type : tableau integer

Obligatoire : non

Liste des codes de sortie qu'Amazon ECS ignorera et ne tentera pas de redémarrer. Vous pouvez spécifier un maximum de 50 codes de sortie de conteneur. Par défaut, Amazon ECS n'ignore aucun code de sortie.

`restartAttemptPeriod`

Type : entier

Obligatoire : non

Période (en secondes) pendant laquelle le conteneur doit fonctionner avant de pouvoir tenter de redémarrer. Un conteneur ne peut être redémarré qu'une fois par `restartAttemptPeriod` seconde. Si un conteneur n'est pas en mesure de fonctionner pendant cette période et qu'il sort plus tôt que prévu, il ne sera pas redémarré. Vous pouvez définir un minimum `restartAttemptPeriod` de 60 secondes et un maximum `restartAttemptPeriod` de 1800 secondes. Par défaut, un conteneur doit fonctionner pendant 300 secondes avant de pouvoir être redémarré.

Surveillance de l'état

`healthCheck`

Commande de surveillance de l'état du conteneur et paramètres de configuration associés pour le conteneur. Pour de plus amples informations, veuillez consulter Déterminer l'état des tâches Amazon ECS à l'aide de vérifications de l'état des conteneurs.

`command`

Tableau de chaînes représentant la commande que le conteneur exécute pour déterminer si celle-ci est saine. Le tableau de chaînes peut commencer par `CMD` pour exécuter directement

les arguments de la commande, ou par `CMD-SHELL` pour exécuter la commande avec le shell par défaut du conteneur. Si vous n'en spécifiez aucun, `CMD` est utilisé.

Lorsque vous enregistrez une définition de tâche dans le AWS Management Console, utilisez une liste de commandes séparées par des virgules. Ces commandes sont converties en une chaîne une fois la définition de tâche créée. Un exemple d'entrée pour une surveillance de l'état est le suivant.

```
CMD-SHELL, curl -f http://localhost/ || exit 1
```

Lorsque vous enregistrez une définition de tâche à l'aide du AWS CLI panneau AWS Management Console JSON APIs, placez la liste des commandes entre crochets. Un exemple d'entrée pour une surveillance de l'état est le suivant.

```
[ "CMD-SHELL", "curl -f http://localhost/ || exit 1" ]
```

Un code de sortie de 0, avec aucune sortie `stderr`, indique la réussite et un code de sortie autre que zéro indique qu'il s'agit d'un échec.

`interval`

La durée (en secondes) entre chaque surveillance de l'état. Vous pouvez indiquer une durée comprise entre 5 et 300 secondes. La valeur par défaut est de 30 secondes.

`timeout`

La durée (en secondes) d'attente pour qu'une surveillance de l'état réussisse avant qu'elle ne soit considérée comme un échec. Vous pouvez indiquer une durée comprise entre 2 et 60 secondes. La valeur par défaut est de 5 secondes.

`retries`

Nombre de nouvelles tentatives d'exécution d'une surveillance de l'état ayant échoué avant que le conteneur soit considéré comme défectueux. Vous pouvez indiquer un nombre de tentatives compris en 1 et 10. La valeur par défaut est 3 nouvelles tentatives.

`startPeriod`

La période de grâce facultative pour donner aux conteneurs le temps de démarrer avant l'échec des surveillance de l'état est prise en compte dans le nombre maximal de tentatives. Vous pouvez indiquer une durée comprise entre 0 et 300 secondes. Par défaut, la `startPeriod` est désactivée.

Si une vérification de l'état aboutit dans le délai défini par `startPeriod`, le conteneur est considéré comme sain et toutes les défaillances suivantes sont prises en compte dans le nombre maximal de nouvelles tentatives.

Environnement

`cpu`

Type : entier

Obligatoire : non

Le nombre d'unités `cpu` que l'agent de conteneur Amazon ECS réserve pour le conteneur. Sous Linux, ce paramètre correspond `CpuShares` à la section Créer un conteneur.

> **Note**
>
> Vous pouvez déterminer le nombre d'unités de processeur disponibles pour chaque type d' EC2 instance Amazon. Pour ce faire, multipliez le nombre de v CPUs répertorié pour ce type d'instance sur la page détaillée EC2des instances Amazon par 1 024.

Les conteneurs Linux partagent des unités de processeur non allouées avec les autres conteneurs de l'instance de conteneur selon le même ratio que la quantité allouée. Par exemple, supposons que vous exécutez une tâche à conteneur unique sur un type d'instance à cœur unique avec 512 unités de processeur spécifiées pour ce conteneur. De plus, cette tâche est la seule tâche en cours d'exécution sur l'instance de conteneur. Dans cet exemple, le conteneur peut utiliser le partage complet de 1 024 unités UC à tout moment. Toutefois, supposons que vous ayez lancé une autre copie de la même tâche sur cette instance de conteneur. Un minimum de 512 unités de processeur est affecté à chaque tâche si nécessaire. De même, si l'autre conteneur n'utilise pas le processeur restant, chaque conteneur peut bénéficier d'une utilisation plus importante du processeur. Toutefois, si les deux tâches sont actives à 100 % en permanence, elles sont limitées à 512 unités d'UC.

Sur les instances de conteneur Linux, le démon Docker de l'instance de conteneur se sert de la valeur de processeur pour calculer les ratios de partage de processeur relatifs pour les conteneurs en cours d'exécution. La valeur de partage de processeur valide minimale autorisée par le noyau Linux est de 2, et la valeur maximale de partage de processeur valide autorisée par

le noyau Linux est de 262 144. Cependant, le paramètre CPU n'est pas obligatoire et vous pouvez utiliser des valeurs de processeur inférieures à deux et supérieures à 262144 dans vos définitions de conteneur. Pour les valeurs de processeur inférieures à deux (y compris null) et supérieures à 262144, le comportement varie en fonction de la version de votre agent de conteneur Amazon ECS :

- Versions de l'agent <= 1.1.0 : les valeurs de processeur nulles et égales à zéro sont transmises à Docker en tant que 0, ce que Docker convertit ensuite en 1 024 parts de processeur. Les valeurs de processeur de 1 sont transmises à Docker en tant que 1, ce que le noyau Linux convertit en deux parts de processeur.
- Versions de l'agent > = 1.2.0 : les valeurs de processeur nulles, 0 et 1 sont transmises à Docker en tant que deux partages de processeur..
- Versions d'agent >= 1.84.0 : les valeurs de processeur supérieures à 256 vCPU sont transmises à 256, ce qui équivaut à Docker 262 144 partages de processeur.

Sur les instances de conteneur Windows, le quota de processeur est appliqué comme quota absolu. Les conteneurs Windows n'ont accès qu'à la quantité spécifiée de processeur décrite dans la définition de tâche. Une valeur de processeur nulle ou égale à zéro est transmise à Docker comme étant `0`. Windows interprète ensuite cette valeur comme 1 % d'un processeur.

Pour voir d'autres exemples, veuillez consulter le billet de blog How Amazon ECS manages CPU and memory resources.

`gpu`

Type : objet ResourceRequirement

Obligatoire : non

Le nombre de `GPUs` physiques que l'agent de conteneur Amazon ECS réserve pour le conteneur. Le nombre de conteneurs GPUs réservés à tous les conteneurs d'une tâche ne doit pas dépasser le nombre de conteneurs disponibles GPUs sur l'instance de conteneur sur laquelle la tâche est lancée. Pour de plus amples informations, veuillez consulter Définitions de tâches Amazon ECS pour les charges de travail du GPU.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

`Elastic Inference accelerator`

Type : objet ResourceRequirement

Obligatoire : non

Pour le type `InferenceAccelerator`, `value` correspond à `deviceName` d'un `InferenceAccelerator` spécifié dans une définition de tâche. Pour de plus amples informations, veuillez consulter the section called “Nom de l'accélérateur Elastic Inference”.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

`essential`

Type : booléen

Obligatoire : non

Supposons que le paramètre `essential` d'un conteneur soit marqué comme `true`, et que ce conteneur échoue ou s'arrête pour une raison quelconque. Ensuite, tous les autres conteneurs qui font partie de la tâche sont arrêtés. Si le paramètre `essential` d'un conteneur a la valeur `false`, son échec n'a pas d'incidence sur le reste des conteneurs dans la tâche. Si ce paramètre n'est pas spécifié, le conteneur est supposé essentiel.

Toutes les tâches doivent comporter au moins un conteneur essentiel. Supposons que vous disposiez d'une application composée de plusieurs conteneurs. Ensuite, les conteneurs de groupes qui sont utilisés dans un même but pour former des composants, et séparer les différents composants en plusieurs définitions de tâches. Pour de plus amples informations, veuillez consulter Architectez votre application pour Amazon ECS.

```
"essential": true|false
```

`entryPoint`

> **Important**
>
> Les premières versions de l'agent de conteneur Amazon ECS ne traitent pas correctement les paramètres `entryPoint`. Si vous rencontrez des difficultés pour utiliser

`entryPoint`, mettez à jour l'agent de conteneur ou saisissez plutôt vos commandes et vos arguments sous forme d'éléments de tableau `command`.

Type : tableau de chaînes

Obligatoire : non

Point d'entrée qui est transmis au conteneur.

```
"entryPoint": ["string", ...]
```

`command`

Type : tableau de chaînes

Obligatoire : non

La commande transmise au conteneur. Ce paramètre correspond `Cmd` à la commande create-container et au `COMMAND` paramètre à docker run. S'il existe plusieurs arguments, assurez-vous que chaque argument est une chaîne séparée dans le tableau.

```
"command": ["string", ...]
```

`workingDirectory`

Type : chaîne

Obligatoire : non

Répertoire de travail dans lequel exécuter les commandes à l'intérieur du conteneur. Ce paramètre correspond à `WorkingDir` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--workdir` correspond à docker run.

```
"workingDirectory": "string"
```

`environmentFiles`

Type : tableau d'objets

Obligatoire : non

Liste des fichiers contenant les variables d'environnement à transmettre à un conteneur. Ce paramètre correspond à l'`--env-file`option de la commande docker run.

Lorsque le protocole FIPS est activé, les noms de compartiments comportant des points (.) (par exemple, amzn-s3-demo-bucket1.name.example) ne sont pas pris en charge. La présence de points (.) dans le nom du compartiment empêche le démarrage de la tâche car l'agent ne peut pas extraire le fichier de variables d'environnement depuis Amazon S3.

Ceci n'est pas disponible pour les conteneurs Windows.

Vous pouvez spécifier jusqu'à dix fichiers d'environnement. Le fichier doit avoir une extension de fichier `.env`. Chaque ligne d'un fichier d'environnement contient une variable d'environnement au format `VARIABLE=VALUE`. Les lignes commençant par # sont traitées comme des commentaires et sont ignorées.

Si des variables d'environnement individuelles sont spécifiées dans la définition du conteneur, elles ont priorité sur les variables contenues dans un fichier d'environnement. Si plusieurs fichiers d'environnement contenant la même variable sont spécifiés, ils sont traités de haut en bas. Nous vous recommandons d'utiliser des noms de variables uniques. Pour de plus amples informations, veuillez consulter Transmettre une variable d'environnement individuelle à un conteneur Amazon ECS.

`value`

Type : String

Obligatoire : oui

L'Amazon Resource Name (ARN) de l'objet Amazon S3 contenant le fichier de variable d'environnement.

`type`

Type : String

Obligatoire : oui

Type de fichier à utiliser La seule valeur prise en charge est `s3`.

`environment`

Type : tableau d'objets

Obligatoire : non

Variables d'environnement à transmettre à un conteneur. Ce paramètre correspond `Env` à la commande docker create-container et à l'option `--env` à la commande docker run.

> **Important**
>
> Nous déconseillons l'utilisation des variables d'environnement en texte clair pour les informations sensibles, comme les données d'identification.

`name`

Type : String

Obligatoire : oui lorsque `environment` est utilisé

Nom de la variable d'environnement.

`value`

Type : String

Obligatoire : oui lorsque `environment` est utilisé

Valeur de la variable d'environnement.

```
"environment" : [
    { "name" : "string", "value" : "string" },
    { "name" : "string", "value" : "string" }
]
```

`secrets`

Type : tableau d'objets

Obligatoire : non

Objet représentant le secret à exposer à votre conteneur. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.

`name`

Type : String

Obligatoire : oui

Valeur à définir comme variable d'environnement sur le conteneur.

`valueFrom`

Type : String

Obligatoire : oui

Secret à exposer au conteneur. Les valeurs prises en charge sont soit le nom Amazon Resource (ARN) complet du AWS Secrets Manager secret, soit l'ARN complet du paramètre dans le AWS Systems Manager Parameter Store.

> **Note**
>
> Si le paramètre Systems Manager Parameter Store ou le paramètre Secrets Manager existe au même endroit Région AWS que la tâche que vous lancez, vous pouvez utiliser l'ARN complet ou le nom du secret. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

```
"secrets": [
    {
        "name": "environment_variable_name",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }
]
```

Paramètres réseau

`disableNetworking`

Type : booléen

Obligatoire : non

Quand ce paramètre a la valeur true, la mise en réseau est désactivée dans le conteneur.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le mode réseau `awsvpc`.

L'argument par défaut est `false`.

```
"disableNetworking": true|false
```

`links`

Type : tableau de chaînes

Obligatoire : non

Le paramètre `link` permet aux conteneurs de communiquer entre eux sans avoir besoin de définir des mappages de ports. Ce paramètre n'est pris en charge que si le mode réseau d'une définition de tâche est défini sur `bridge`. La construction `name:internalName` est analogue à `name:alias` dans les liaisons Docker. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits d'union et traits de soulignement sont autorisés.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le mode réseau `awsvpc`.

> **Important**
>
> Des conteneurs situés sur la même instance de conteneur peuvent communiquer entre eux sans nécessiter de liens ou de mappages de ports hôtes. L'isolation réseau sur une instance de conteneur est contrôlée par les groupes de sécurité et les paramètres de VPC.

```
"links": ["name:internalName", ...]
```

`hostname`

Type : chaîne

Obligatoire : non

Nom d'hôte à utiliser pour votre conteneur. Ce paramètre correspond `Hostname` au docker create-container et à l'option `--hostname` docker run.

> **Note**
>
> Si vous utilisez le mode réseau `awsvpc`, le paramètre `hostname` n'est pas pris en charge.

```
"hostname": "string"
```

`dnsServers`

Type : tableau de chaînes

Obligatoire : non

Liste de serveurs DNS qui sont présentés au conteneur.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le mode réseau `awsvpc`.

```
"dnsServers": ["string", ...]
```

`dnsSearchDomains`

Type : tableau de chaînes

Obligatoire : non

Modèle : ^[a-zA-Z0-9-.]{0,253}[a-zA-Z0-9]$

Liste des domaines de recherche DNS qui sont présentés au conteneur. Ce paramètre correspond à l'option `--dns-search` docker run `DnsSearch` dans la commande docker create-container.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le mode réseau `awsvpc`.

```
"dnsSearchDomains": ["string", ...]
```

`extraHosts`

Type : tableau d'objets

Obligatoire : non

Liste de noms d'hôte et de mappages d'adresses IP à ajouter au fichier `/etc/hosts` dans le conteneur.

Ce paramètre correspond `ExtraHosts` à la commande docker create-container et à l'option `--add-host` docker run.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le mode réseau `awsvpc`.

```
"extraHosts": [
      {
        "hostname": "string",
        "ipAddress": "string"
      }
      ...
    ]
```

`hostname`

Type : String

Obligatoire : oui, lorsque des objets `extraHosts` sont utilisés

Nom d'hôte à utiliser dans l'entrée `/etc/hosts`.

`ipAddress`

Type : String

Obligatoire : oui, lorsque des objets `extraHosts` sont utilisés

Adresse IP à utiliser dans l'entrée `/etc/hosts`.

Stockage et journalisation

`readonlyRootFilesystem`

Type : booléen

Obligatoire : non

Quand ce paramètre a la valeur true, le conteneur ne dispose que d'un accès en lecture seule au système de fichiers racine. Ce paramètre correspond à l'option `--read-only` docker run `ReadonlyRootfs` dans la commande docker create-container.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

L'argument par défaut est `false`.

```
"readonlyRootFilesystem": true|false
```

`mountPoints`

Type : tableau d'objets

Obligatoire : non

Les points de montage des volumes de données de votre conteneur. Ce paramètre correspond `Volumes` à l'API Docker create-container et `--volume` à l'option docker run.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`. Les conteneurs Windows ne peuvent pas monter de répertoires sur un autre lecteur, et les points de montage ne peuvent pas être utilisés sur plusieurs lecteurs. Vous devez spécifier des points de montage pour associer un volume Amazon EBS directement à une tâche Amazon ECS.

`sourceVolume`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Le chemin dans le conteneur où le volume sera monté.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

Pour les tâches qui exécutent le système d'exploitation Windows, laissez la valeur par défaut de`false`.

`volumesFrom`

Type : tableau d'objets

Obligatoire : non

Volumes de données à monter à partir d'un autre conteneur. Ce paramètre correspond `VolumesFrom` à la commande docker create-container et à l'option `--volumes-from` docker run.

`sourceContainer`

Type : String

Obligatoire : oui lorsque `volumesFrom` est utilisé

Nom du conteneur à partir duquel monter les volumes.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

```
"volumesFrom": [
                {
                  "sourceContainer": "string",
                  "readOnly": true|false
                }
              ]
```

`logConfiguration`

Type : LogConfigurationObjet

Obligatoire : non

Spécification de configuration du journal pour le conteneur.

Pour obtenir des exemples de définitions de tâche qui utilisent une configuration de journaux, consultez Exemples de définitions de tâches Amazon ECS.

Ce paramètre correspond `LogConfig` à la commande docker create-container et à l'option `--log-driver` docker run. Par défaut, les conteneurs utilisent le même pilote de journalisation que celui utilisé par le démon Docker. Cependant, le conteneur peut utiliser un pilote de journalisation différent que celui du démon Docker en spécifiant un pilote de journal avec ce paramètre dans la définition de conteneur. Pour utiliser un pilote de journalisation différent pour un conteneur, le système de journal doit être configuré correctement sur l'instance de conteneur (ou sur un serveur de journal différent pour les options de journalisation à distance).

Tenez compte des éléments suivants lorsque vous spécifiez une configuration de journal pour vos conteneurs :

- Amazon ECS prend en charge un sous-ensemble des pilotes de journalisation qui sont disponibles pour le démon Docker. Les pilotes de journal supplémentaires qui ne sont pas pris en charge actuellement seront peut-être disponibles dans les prochaines versions de l'agent de conteneur Amazon ECS.
- Ce paramètre nécessite la version 1.18 ou ultérieure de l'API Docker à distance sur votre instance de conteneur.
- L'agent de conteneur Amazon ECS qui s'exécute sur une instance de conteneur doit enregistrer les pilotes de journalisation disponibles sur cette instance avec la variable

d'`ECS_AVAILABLE_LOGGING_DRIVER`Environnement avant que les conteneurs placés sur cette instance puissent utiliser ces options de configuration de journal. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

```
"logConfiguration": {
      "logDriver": "awslogs","fluentd","gelf","json-
file","journald","splunk","syslog","awsfirelens",
      "options": {"string": "string"
        ...},
 "secretOptions": [{
  "name": "string",
  "valueFrom": "string"
 }]
}
```

`logDriver`

Type : String

Valeurs valides : `"awslogs","fluentd","gelf","json-file","journald","splunk","syslog","awsfirelens"`

Obligatoire : oui lorsque `logConfiguration` est utilisé

Pilote de journal à utiliser pour le conteneur. Par défaut, les valeurs valides mentionnées ci-dessus sont les pilotes de journal avec lesquels l'agent de conteneur Amazon ECS peut communiquer.

Les pilotes de journal pris en charge sont `awslogs`, `fluentd`, `gelf`, `json-file`, `journald`, `syslog`, `splunk` et `awsfirelens`.

Pour plus d'informations sur l'utilisation du pilote de `awslogs` journal dans les définitions de tâches pour envoyer les journaux de vos conteneurs à CloudWatch Logs, consultezEnvoyez les journaux Amazon ECS à CloudWatch .

Pour de plus amples informations sur l'utilisation du pilote de journal `awsfirelens`, consultez Routage personnalisé des journaux.

> **Note**
>
> Si vous avez un pilote personnalisé qui n'est pas répertorié, vous pouvez bifurquer le projet d'agent de conteneur Amazon ECS disponible sur GitHub et le personnaliser

pour qu'il fonctionne avec ce pilote. Nous vous conseillons d'envoyer des demandes d'extraction pour les modifications que vous souhaitez inclure. Toutefois, nous ne prenons pas en charge l'exécution de copies modifiées de ce logiciel pour le moment.

Ce paramètre nécessite la version 1.18 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

`options`

Type : mappage chaîne/chaîne

Obligatoire : non

key/value Carte des options de configuration à envoyer au pilote du journal.

Les options que vous pouvez spécifier dépendent du pilote de journal. Certaines des options que vous pouvez spécifier lorsque vous utilisez le `awslogs` routeur pour acheminer les journaux vers Amazon CloudWatch sont les suivantes :

`awslogs-create-group`

Obligatoire : non

Spécifiez si vous souhaitez que le groupe de journaux soit créé automatiquement. Si cette option n'est pas spécifiée, l'emplacement par défaut est `false`.

> **Note**
>
> Votre politique IAM doit inclure l'autorisation `logs:CreateLogGroup` afin que vous puissiez essayer d'utiliser `awslogs-create-group`.

`awslogs-region`

Obligatoire : oui

Spécifiez Région AWS le pilote de `awslogs` journal auquel envoyer vos journaux Docker. Vous pouvez choisir d'envoyer tous vos journaux provenant de clusters situés dans différentes régions vers une seule région dans CloudWatch Logs. Cela leur permet d'être tous visibles en un seul endroit. Sinon, vous pouvez les séparer par région pour plus de

granularité. Assurez-vous que le groupe de journaux spécifié existe dans la région que vous spécifiez avec cette option.

`awslogs-group`

Obligatoire : oui

Assurez-vous de spécifier un groupe de journaux auquel le pilote de journaux `awslogs` envoie ses flux de journaux.

`awslogs-stream-prefix`

Obligatoire : Facultatif

Utilisez l'option `awslogs-stream-prefix` pour associer un flux de journaux au préfixe spécifié, au nom du conteneur et à l'ID de la tâche Amazon ECS à laquelle appartient le conteneur. Si vous spécifiez un préfixe avec cette option, le format du flux de journaux est le suivant.

```
prefix-name/container-name/ecs-task-id
```

Si vous ne spécifiez pas de préfixe avec cette option, le flux de journaux est nommé d'après l'ID de conteneur attribué par le démon Docker sur l'instance de conteneur. Etant donné qu'il est difficile de suivre les journaux jusqu'au conteneur qui les a envoyé en utilisant uniquement l'ID de conteneur Docker (qui est uniquement disponible sur l'instance de conteneur), nous vous recommandons de spécifier un préfixe avec cette option.

Pour les services Amazon ECS, vous pouvez utiliser le nom du service comme préfixe. Ce faisant, vous pourrez suivre les flux de journaux et déterminer le service auquel appartient le conteneur, trouver le nom du conteneur qui les a envoyés, ainsi que l'ID de la tâche à laquelle le conteneur appartient.

Vous devez spécifier un préfixe de flux pour vos journaux pour qu'ils apparaissent dans le panneau des journaux lors de l'utilisation de la console Amazon ECS.

`awslogs-datetime-format`

Obligatoire : non

Cette option définit un modèle de démarrage à plusieurs lignes au format `strftime` Python. Un message de journal est composé de plusieurs lignes : une ligne correspondant

au modèle et les lignes suivantes qui ne correspondent pas au modèle. La ligne mise en correspondance est le délimiteur entre les messages de journalisation.

Ce format peut, par exemple, servir à analyser une sortie comme une pile de vidage, laquelle pourrait, dans le cas contraire, être consignée en plusieurs entrées. Le modèle adéquat permet de la capturer dans une seule entrée.

Pour de plus amples informations, veuillez consulter awslogs-datetime-format.

Vous ne pouvez pas configurer à la fois les options `awslogs-datetime-format` et `awslogs-multiline-pattern`.

> **Note**
>
> La journalisation multiligne effectue l'analyse et la mise en correspondance des expressions régulières de tous les messages de journalisation. Cela peut avoir un impact négatif sur les performances de journalisation.

`awslogs-multiline-pattern`

Obligatoire : non

Cette option définit un modèle de démarrage à plusieurs lignes à l'aide d'une expression régulière. Un message de journal est composé de plusieurs lignes : une ligne correspondant au modèle et les lignes suivantes qui ne correspondent pas au modèle. La ligne mise en correspondance est le délimiteur entre les messages de journalisation.

Pour de plus amples informations, veuillez consulter awslogs-multiline-pattern.

Cette option est ignorée si `awslogs-datetime-format` est également configuré.

Vous ne pouvez pas configurer à la fois les options `awslogs-datetime-format` et `awslogs-multiline-pattern`.

> **Note**
>
> La journalisation multiligne effectue l'analyse et la mise en correspondance des expressions régulières de tous les messages de journalisation. Cela peut avoir un impact négatif sur les performances de journalisation.

Les options suivantes s'appliquent à tous les pilotes de journal pris en charge.

`mode`

Obligatoire : non

Valeurs valides : `non-blocking` | `blocking`

Cette option définit le mode de livraison des messages de journal du conteneur au pilote de journal spécifié à l'aide de`logDriver`. Le mode de livraison que vous choisissez affecte la disponibilité des applications lorsque le flux de logs provenant du conteneur est interrompu.

Si vous utilisez ce `blocking` mode et que le flux de logs est interrompu, les appels provenant du code du conteneur pour écrire dans les `stderr` flux `stdout` et seront bloqués. Le fil de journalisation de l'application se bloquera en conséquence. Cela peut empêcher l'application de répondre et entraîner l'échec de vérification de l'état du conteneur.

Si vous utilisez le mode `non-blocking`, les journaux du conteneur sont plutôt stockés dans une mémoire tampon intermédiaire configurée avec l'option `max-buffer-size`. Cela empêche l'application de ne plus répondre lorsque les journaux ne peuvent pas être envoyés. Nous vous recommandons d'utiliser ce mode si vous souhaitez garantir la disponibilité du service et si vous acceptez une certaine perte de journaux. Pour plus d'informations, consultez la section Prévention de la perte de journal grâce au mode non bloquant dans le pilote de journal du `awslogs` conteneur.

Vous pouvez définir une valeur par défaut `mode` pour tous les conteneurs d'un conteneur spécifique en Région AWS utilisant les paramètres du `defaultLogDriverMode` compte. Si vous ne spécifiez pas l'`mode`option dans les paramètres du compte `logConfiguration` ou si vous ne configurez pas le paramètre, Amazon ECS passera en `non-blocking` mode par défaut. Pour plus d'informations sur le paramétrage du compte, consultezMode pilote de journal par défaut.

Quand le mode `non-blocking` est utilisé, l'option de journalisation `max-buffer-size` contrôle la taille de la mémoire tampon utilisée pour le stockage des messages intermédiaires. Assurez-vous de spécifier une taille de mémoire tampon adéquate en fonction de votre application. La quantité totale de mémoire allouée au niveau de la tâche doit être supérieure à la quantité de mémoire allouée à tous les conteneurs en plus de la mémoire tampon du pilote de journal.

> **Note**
>
> Le 25 juin 2025, Amazon ECS a modifié le mode pilote de journal par défaut de à pour donner la priorité `blocking` `non-blocking` à la disponibilité des tâches plutôt qu'à la journalisation. Pour continuer à utiliser le `blocking` mode après cette modification, effectuez l'une des opérations suivantes :
>
> - Définissez l'`mode`option dans votre définition de conteneur `logConfiguration` comme`blocking`.
> - Réglez le paramètre du `defaultLogDriverMode` compte sur`blocking`.

`max-buffer-size`

Obligatoire : non

Valeur par défaut : `10 m`

Quand le mode `non-blocking` est utilisé, l'option de journalisation `max-buffer-size` contrôle la taille de la mémoire tampon utilisée pour le stockage des messages intermédiaires. Assurez-vous de spécifier une taille de mémoire tampon adéquate en fonction de votre application. Lorsque la mémoire tampon est pleine, il est impossible de stocker d'autres journaux. Les journaux qui ne peuvent pas être stockés sont perdus.

Pour acheminer les journaux à l'aide du `splunk` routeur de journaux, vous devez spécifier a `splunk-token` et `splunk-url` a.

Lorsque vous utilisez le routeur de `awsfirelens` journaux pour acheminer les journaux vers une AWS Partner Network destination Service AWS ou à des fins de stockage et d'analyse des journaux, vous pouvez définir l'`log-driver-buffer-limit`option permettant de limiter le nombre d'événements mis en mémoire tampon avant d'être envoyés au conteneur du routeur de journaux. Cela peut aider à résoudre le problème potentiel de perte de journal, car un débit élevé peut entraîner un épuisement de la mémoire pour le tampon à l'intérieur de Docker. Pour de plus amples informations, veuillez consulter the section called “Configuration des journaux pour un débit élevé”.

Les autres options que vous pouvez spécifier lors de l'utilisation `awsfirelens` pour acheminer les journaux dépendent de la destination. Lorsque vous exportez des journaux vers Amazon Data Firehose, vous pouvez spécifier le Région AWS format `region` et le nom du flux de journaux. `delivery_stream`

Lorsque vous exportez des journaux vers Amazon Kinesis Data Streams, vous pouvez spécifier Région AWS un `region` with et un nom `stream` de flux de données avec.

Lorsque vous exportez des journaux vers Amazon OpenSearch Service, vous pouvez spécifier des options telles que `Name` `Host` (point de terminaison de OpenSearch service sans protocole) `Port` `IndexType`,`Aws_auth`,`Aws_region`,,`Suppress_Type_Name`, et`tls`.

Lorsque vous exportez des journaux vers Amazon S3, vous pouvez spécifier le compartiment à l'aide de l'`bucket`option. Vous pouvez également spécifier`region`, `total_file_size``upload_timeout`, et `use_put_object` en tant qu'options.

Ce paramètre nécessite la version 1.19 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

`secretOptions`

Type : tableau d'objets

Obligatoire : non

Objet qui représente le secret à transmettre à la configuration de journal. Les secrets utilisés dans la configuration du journal peuvent inclure un jeton d'authentification, un certificat ou une clé de chiffrement. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.

`name`

Type : String

Obligatoire : oui

Valeur à définir comme variable d'environnement sur le conteneur.

`valueFrom`

Type : String

Obligatoire : oui

Secret à exposer à la configuration de journal du conteneur.

```
"logConfiguration": {
 "logDriver": "splunk",
```

```
  "options": {
   "splunk-url": "https://cloud.splunk.com:8080",
   "splunk-token": "...",
   "tag": "...",
   ...
  },
  "secretOptions": [{
   "name": "splunk-token",
   "valueFrom": "/ecs/logconfig/splunkcred"
  }]
 }
```

`firelensConfiguration`

Type : FirelensConfigurationObjet

Obligatoire : non

La FireLens configuration du conteneur. Cette option est utilisée pour spécifier et configurer un routeur de journal pour les journaux de conteneur. Pour de plus amples informations, veuillez consulter Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

```
{
    "firelensConfiguration": {
        "type": "fluentd",
        "options": {
            "KeyName": ""
        }
    }
}
```

`options`

Type : mappage chaîne/chaîne

Obligatoire : non

key/value Carte des options à utiliser lors de la configuration du routeur de journalisation. Ce champ est facultatif et peut être utilisé pour spécifier un fichier de configuration personnalisé ou ajouter des métadonnées supplémentaires, telles que les détails de tâche, de définition de tâche, de cluster et d'instance de conteneur à l'événement de journal. Si spécifié, la syntaxe à utiliser est `"options":{"enable-ecs-log-metadata":"true|false","config-`

`file-type:"s3|file","config-file-value":"arn:aws:s3:::amzn-s3-demo-bucket/fluent.conf|filepath"}`. Pour de plus amples informations, veuillez consulter Exemple de définition de tâche Amazon ECS : acheminer les journaux vers FireLens.

`type`

Type : String

Obligatoire : oui

Routeur de journal à utiliser. Les valeurs valides sont `fluentd` ou `fluentbit`.

Sécurité

Pour plus d'informations sur la sécurité des conteneurs, consultez les meilleures pratiques en matière de sécurité des conteneurs et des tâches Amazon ECS.

`credentialSpecs`

Type : tableau de chaînes

Obligatoire : non

Liste des informations contenues ARNs dans SSM ou Amazon S3 dans un fichier de spécifications d'identification (`CredSpec`) qui configure le conteneur pour l'authentification Active Directory. Nous vous recommandons d'utiliser ce paramètre plutôt que les `dockerSecurityOptions`. Le nombre maximum de ARNs est 1.

Il existe deux formats pour chaque ARN.

credentialspecdomainless:MyARN

Vous utilisez `credentialspecdomainless:MyARN` pour fournir un `CredSpec` avec une section supplémentaire pour un secret dans Secrets Manager. Vous fournissez les informations d'identification de connexion au domaine dans le secret.

Chaque tâche exécutée sur une instance de conteneur peut rejoindre différents domaines.

Vous pouvez utiliser ce format sans associer l'instance de conteneur à un domaine.

credentialspec:MyARN

Vous utilisez `credentialspec:MyARN` pour fournir un `CredSpec` pour un domaine unique.

Vous devez joindre l'instance de conteneur au domaine avant de démarrer toute tâche utilisant cette définition de tâche.

Dans les deux formats, remplacez `MyARN` par l'ARN dans SSM ou Amazon S3.

Le `credspec` doit fournir un ARN dans Secrets Manager pour un secret contenant le nom d'utilisateur, le mot de passe et le domaine auquel se connecter. Pour une meilleure sécurité, l'instance n'est pas jointe au domaine pour l'authentification sans domaine. Les autres applications de l'instance ne peuvent pas utiliser les informations d'identification sans domaine. Vous pouvez utiliser ce paramètre pour exécuter des tâches sur la même instance, même si les tâches doivent être jointes à différents domaines. Pour plus d'informations, consultez les sections Utilisation de g MSAs pour les conteneurs Windows et Utilisation de g MSAs pour les conteneurs Linux.

`privileged`

Type : booléen

Obligatoire : non

Quand ce paramètre a la valeur true, le conteneur dispose de droits élevés sur l'instance de conteneur hôte (similaire à l'utilisateur `root`). Nous vous déconseillons d'exécuter des conteneurs avec `privileged`. Dans la plupart des cas, vous pouvez spécifier les privilèges exacts dont vous avez besoin en utilisant les paramètres spécifiques au lieu d'utiliser `privileged`.

Ce paramètre correspond `Privileged` à la commande docker create-container et à l'option `--privileged` docker run.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le type de lancement Fargate.

L'argument par défaut est `false`.

```
"privileged": true|false
```

`user`

Type : chaîne

Obligatoire : non

Utilisateur à utiliser à l'intérieur du conteneur. Ce paramètre correspond `User` à la commande docker create-container et à l'option `--user` docker run.

> **Important**
>
> Lorsque vous exécutez des tâches en mode réseau `host`, n'exécutez pas de conteneurs à l'aide de l'utilisateur root (UID 0). Comme bonne pratique de sécurité, utilisez toujours un utilisateur non root.

Vous pouvez spécifier l'`user` à l'aide des formats suivants. Si vous spécifiez un UID ou GID, vous devez le spécifier en tant que nombre entier positif.

- `user`
- `user:group`
- `uid`
- `uid:gid`
- `user:gid`
- `uid:group`

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"user": "string"
```

`dockerSecurityOptions`

Type : tableau de chaînes

Valeurs valides : « » | no-new-privileges « apparmor:profile » | « label : » | « *`value`* credentialspec : » *`CredentialSpecFilePath`*

Obligatoire : non

Liste de chaînes pour fournir une configuration personnalisée pour plusieurs systèmes de sécurité.

Pour les Linux tâches, ce paramètre peut être utilisé pour référencer des étiquettes personnalisées SELinux et des systèmes de sécurité à AppArmor plusieurs niveaux.

Ce paramètre peut être utilisé pour référencer un fichier de spécifications d'identification qui configure un conteneur pour l'authentification Active Directory. Pour plus d'informations, consultez Découvrez comment utiliser les conteneurs g MSAs pour EC2 Windows pour Amazon ECS et Utilisation gMSA pour les EC2 Linux conteneurs sur Amazon ECS.

Ce paramètre correspond `SecurityOpt` à la commande docker create-container et à l'option `--security-opt` docker run.

```
"dockerSecurityOptions": ["string", ...]
```

> **Note**
>
> L'agent de conteneur Amazon ECS en cours d'exécution sur une instance de conteneur doit s'enregistrer avec les variables d'environnement `ECS_SELINUX_CAPABLE=true` ou `ECS_APPARMOR_CAPABLE=true` pour que les conteneurs placés sur cette instance puissent utiliser ces options de sécurité. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

Limites des ressources

`ulimits`

Type : tableau d'objets

Obligatoire : non

Une liste de valeurs `ulimit` à définir pour un conteneur. Cette valeur remplace le paramètre de quota de ressources par défaut pour le système d'exploitation. Ce paramètre correspond `Ulimits` à la commande docker create-container et à l'option `--ulimit` docker run.

Ce paramètre nécessite la version 1.18 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"ulimits": [
      {
        "name":
 "core"|"cpu"|"data"|"fsize"|"locks"|"memlock"|"msgqueue"|"nice"|"nofile"|"nproc"|"rss"|"rtp
        "softLimit": integer,
        "hardLimit": integer
      }
      ...
    ]
```

`name`

Type : String

Valeurs valides : `"core" | "cpu" | "data" | "fsize" | "locks" | "memlock" | "msgqueue" | "nice" | "nofile" | "nproc" | "rss" | "rtprio" | "rttime" | "sigpending" | "stack"`

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

Le `type` d'`ulimit`.

`hardLimit`

Type : entier

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

La limite stricte du type `ulimit`. La valeur peut être spécifiée en octets, en secondes ou en tant que nombre, selon le `typeulimit`.

`softLimit`

Type : entier

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

La limite flexible du type `ulimit`. La valeur peut être spécifiée en octets, en secondes ou en tant que nombre, selon le `typeulimit`.

Étiquettes Docker

`dockerLabels`

Type : mappage chaîne/chaîne

Obligatoire : non

Une key/value carte des étiquettes à ajouter au conteneur. Ce paramètre correspond `Labels` à la commande docker create-container et à l'option `--label` docker run.

Ce paramètre nécessite la version 1.18 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

```
"dockerLabels": {"string": "string"
      ...}
```

Autres paramètres de définition de conteneur

Les paramètres de définition de conteneur suivants peuvent être utilisés lors de l'enregistrement des définitions de tâche dans la console Amazon ECS à l'aide de l'option Configure via JSON (Configurer via JSON). Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.

Rubriques

- Paramètres Linux
- Dépendances du conteneur
- Temporisations de conteneurs
- Contrôles système
- Interactive
- Pseudo Terminal

Paramètres Linux

`linuxParameters`

Type : objet LinuxParameters

Obligatoire : non

Linux-des options spécifiques appliquées au conteneur, telles KernelCapabilitiesque.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"linuxParameters": {
      "capabilities": {
        "add": ["string", ...],
        "drop": ["string", ...]
        }
      }
```

`capabilities`

Type : objet KernelCapabilities

Obligatoire : non

Caractéristiques Linux du conteneur ajoutées ou supprimées de la configuration par défaut fournie par Docker. Pour plus d'informations sur ces caractéristiques Linux, veuillez consulter la page consacrée aux caractéristiques(7) dans le manuel Linux (langue française non garantie).

`add`

Type : tableau de chaînes

Valeurs valides : `"ALL" | "AUDIT_CONTROL" | "AUDIT_READ" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" | "NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW" | "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" | "SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT" | "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" | "SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"`

Obligatoire : non

Caractéristiques Linux du conteneur à ajouter à la configuration par défaut fournie par Docker. Ce paramètre correspond `CapAdd` à la commande docker create-container et à l'option `--cap-add` docker run.

`add`

Type : tableau de chaînes

Valeurs valides : `"SYS_PTRACE"`

Obligatoire : non

Caractéristiques Linux du conteneur à ajouter à la configuration par défaut fournie par Docker. Ce paramètre correspond `CapAdd` à la commande docker create-container et à l'option `--cap-add` docker run.

`drop`

Type : tableau de chaînes

Valeurs valides : `"ALL" | "AUDIT_CONTROL" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" | "NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW" | "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" | "SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT" | "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" | "SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"`

Obligatoire : non

Caractéristiques Linux du conteneur à supprimer de la configuration par défaut fournie par Docker. Ce paramètre correspond `CapDrop` à la commande docker create-container et à l'option `--cap-drop` docker run.

`devices`

Tous les périphériques hôtes à exposer au conteneur. Ce paramètre correspond `Devices` à la commande docker create-container et à l'option `--device` docker run.

Type : tableau d'objets Périphérique

Obligatoire : non

`hostPath`

Chemin du périphérique dans l'instance de conteneur hôte.

Type : String

Obligatoire : oui

`containerPath`

Chemin auquel exposer l'appareil hôte à l'intérieur du conteneur.

Type : chaîne

Obligatoire : non

`permissions`

Autorisations explicites à fournir au conteneur pour le périphérique. Par défaut, le conteneur dispose des autorisations `read`, `write` et `mknod` sur l'appareil.

Type : tableau de chaînes

Valeurs valides : `read` | `write` | `mknod`

`initProcessEnabled`

Exécutez un processus `init` dans le conteneur afin de transmettre des signaux et de récolter les processus. Ce paramètre correspond à l'`--init`option docker run.

Ce paramètre nécessite la version 1.25 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

`maxSwap`

Quantité totale de mémoire d'échange (en Mio) qu'un conteneur peut utiliser. Ce paramètre est traduit en `--memory-swap` option docker run où la valeur est la somme de la mémoire du conteneur plus la `maxSwap` valeur.

Si la valeur `0` est spécifiée pour `maxSwap`, le conteneur n'utilise pas l'échange. Les valeurs acceptées sont `0` ou n'importe quel nombre entier positif. Si le paramètre `maxSwap` n'est

pas spécifié, le conteneur utilise la configuration d'échange pour l'instance de conteneur sur laquelle il s'exécute. Une valeur `maxSwap` doit être définie pour que le paramètre `swappiness` soit utilisé.

`sharedMemorySize`

Valeur correspondant à la taille (en Mio) du volume `/dev/shm`. Ce paramètre correspond à l'`--shm-size`option docker run.

Type : entier

`swappiness`

Vous pouvez utiliser ce paramètre pour régler le comportement d'échange de mémoire d'un conteneur. Une valeur `swappiness` de `0` empêche l'échange de se produire à moins que cela ne soit nécessaire. Une valeur `swappiness` de `100` entraîne fréquemment l'échange de pages. Les valeurs acceptées sont les nombres entiers compris entre `0` et `100`. Si vous ne spécifiez aucune valeur, la valeur par défaut `60` est utilisée. De plus, si vous ne spécifiez pas de valeur pour `maxSwap`, ce paramètre est ignoré. Ce paramètre correspond à l'`--memory-swappiness`option docker run.

> **Note**
>
> Si vous utilisez des tâches sur Amazon Linux 2023, le paramètre `swappiness` n'est pas pris en charge.

`tmpfs`

Chemin du conteneur, options de montage et taille maximale (en Mio) du montage tmpfs. Ce paramètre correspond à l'`--tmpfs`option docker run.

Type : tableau d'objets Tmpfs

Obligatoire : non

`containerPath`

Chemin de fichier absolu où sera monté le volume tmpfs.

Type : String

Obligatoire : oui

`mountOptions`

Liste des options de montage du volume tmpfs.

Type : tableau de chaînes

Obligatoire : non

Valeurs valides : `"defaults" | "ro" | "rw" | "suid" | "nosuid" | "dev" | "nodev" | "exec" | "noexec" | "sync" | "async" | "dirsync" | "remount" | "mand" | "nomand" | "atime" | "noatime" | "diratime" | "nodiratime" | "bind" | "rbind" | "unbindable" | "runbindable" | "private" | "rprivate" | "shared" | "rshared" | "slave" | "rslave" | "relatime" | "norelatime" | "strictatime" | "nostrictatime" | "mode" | "uid" | "gid" | "nr_inodes" | "nr_blocks" | "mpol"`

`size`

Taille maximale (en Mio) du volume tmpfs.

Type : entier

Obligatoire : oui

Dépendances du conteneur

`dependsOn`

Type : tableau d'objets ContainerDependency

Obligatoire : non

Dépendances définies pour le démarrage et l'arrêt de conteneurs. Un conteneur peut contenir plusieurs dépendances. Lorsqu'une dépendance est définie pour le démarrage des conteneurs, elle est inversée pour leur arrêt. Pour obtenir un exemple, consultez Dépendances du conteneur.

> **Note**
>
> Si un conteneur ne respecte pas une contrainte de dépendance ou s'écoule avant de respecter la contrainte, Amazon ECS ne fait pas progresser pas les conteneurs dépendants vers leur état suivant.

Les instances nécessitent au moins une version `1.26.0` de l'agent de conteneur pour activer les dépendances de conteneur. Cependant, nous vous recommandons d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS. Si vous utilisez l'AMI Linux Amazon optimisée pour Amazon ECS, votre instance doit au moins disposer de la version `1.26.0-1` du package `ecs-init`. Si vos instances de conteneur sont lancées à partir de la version `20190301` ou ultérieure, elles contiennent les versions requises de l'agent de conteneur et `ecs-init`. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.

```
"dependsOn": [
    {
        "containerName": "string",
        "condition": "string"
    }
]
```

`containerName`

Type : String

Obligatoire : oui

Nom de conteneur qui doit répondre à la condition spécifiée.

`condition`

Type : String

Obligatoire : oui

Condition de dépendance du conteneur. Voici les conditions disponibles et leur comportement :

- `START` - Cette condition émule le comportement de liens et de volumes aujourd'hui. La condition vérifie qu'un conteneur dépendant est démarré avant d'autoriser d'autres conteneurs à démarrer.
- `COMPLETE` - Cette condition vérifie qu'un conteneur dépendant exécute un cycle complet (sorties) avant d'autoriser d'autres conteneurs à démarrer. Cela peut être utile pour les conteneurs non essentiels qui exécutent un script, puis se ferment. Cette condition ne peut pas être définie sur un conteneur essentiel.

- `SUCCESS` - Cette condition est identique à `COMPLETE`, mais elle nécessite également que le conteneur se termine par un état `zero`. Cette condition ne peut pas être définie sur un conteneur essentiel.
- `HEALTHY` - Cette condition vérifie que la surveillance de l'état du conteneur dépendant réussit avant d'autoriser d'autres conteneurs à démarrer. Cela nécessite que le conteneur dépendant dispose de surveillances de l'état configurées dans la définition de la tâche. Cette condition est confirmée uniquement au démarrage de tâche.

Temporisations de conteneurs

`startTimeout`

Type : entier

Obligatoire : non

Exemples de valeur : `120`

Durée d'attente (en secondes) avant l'abandon de la résolution de dépendances pour un conteneur.

Par exemple, vous spécifiez deux conteneurs dans une définition de tâche, où le `containerA` dispose d'une dépendance au `containerB` avec un état `COMPLETE`, `SUCCESS` ou `HEALTHY`. Si une valeur `startTimeout` est spécifiée pour `containerB` et qu'il n'a pas atteint l'état souhaité dans ce délai, alors le `containerA` ne démarre pas.

> **Note**
>
> Si un conteneur ne respecte pas une contrainte de dépendance ou s'écoule avant de respecter la contrainte, Amazon ECS ne fait pas progresser pas les conteneurs dépendants vers leur état suivant.

La valeur maximale est de 120 secondes.

`stopTimeout`

Type : entier

Obligatoire : non

Exemples de valeur : `120`

Durée (en secondes) à attendre avant que le conteneur soit expressément tué s'il ne s'achève pas normalement tout seul.

Si le `stopTimeout` paramètre n'est pas spécifié, la valeur définie pour la variable de configuration de l'agent de conteneur Amazon ECS `ECS_CONTAINER_STOP_TIMEOUT` est utilisée. Si ni le paramètre `stopTimeout` ni la variable de configuration de l'agent `ECS_CONTAINER_STOP_TIMEOUT` ne sont définis, les valeurs par défaut de 30 secondes pour les conteneurs Linux et de 30 secondes pour les conteneurs Windows sont utilisées. Les instances de conteneur doivent au moins disposer de la version 1.26.0 de l'agent de conteneur pour permettre une valeur de temporisation d'arrêt d'un conteneur. Cependant, nous vous recommandons d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS. Si vous utilisez l'AMI Amazon Linux optimisée pour Amazon ECS, votre instance doit au moins disposer de la version 1.26.0-1 du package `ecs-init`. Si vos instances de conteneur sont lancées à partir de la version `20190301` ou ultérieure, elles contiennent les versions requises de l'agent de conteneur et `ecs-init`. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.

Contrôles système

`systemControls`

Type : objet SystemControl

Obligatoire : non

Liste des paramètres du noyau de l'espace de noms à définir dans le conteneur. Ce paramètre correspond `Sysctls` à la commande docker create-container et à l'option `--sysctl` docker run. Par exemple, vous pouvez configurer le paramètre `net.ipv4.tcp_keepalive_time` pour maintenir des connexions de plus longue durée.

Il n'est pas recommandé de spécifier des paramètres `systemControls` liés au réseau pour plusieurs conteneurs dans une seule tâche qui utilise également le mode réseau `awsvpc` ou `host`. En procédant ainsi, vous vous exposez aux inconvénients suivants :

- Pour les tâches qui utilisent le mode réseau `awsvpc`, si vous définissez le paramètre `systemControls` pour un conteneur, il s'applique à tous les conteneurs de la tâche. Si

vous définissez différents paramètres `systemControls` pour plusieurs conteneurs dans une seule tâche, c'est le conteneur qui est démarré en dernier qui détermine le paramètre `systemControls` qui prend effet.

- Les définitions de tâche qui utilisent le mode réseau `host`, l'espace de noms `systemControls` ne sont pas prises en charge.

Si vous définissez un espace de noms des ressources IPC à utiliser pour les conteneurs de la tâche, les conditions suivantes s'appliquent aux contrôles de système. Pour de plus amples informations, veuillez consulter Mode IPC.

- Pour les tâches qui utilisent le mode IPC `host`, les `systemControls` liés à l'espace de noms IPC ne sont pas pris en charge.
- Pour les tâches qui utilisent le mode IPC `task`, les valeurs des `systemControls` liés à l'espace de noms IPC s'appliquent à tous les conteneurs au sein d'une tâche.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"systemControls": [
    {
         "namespace":"string",
         "value":"string"
    }
]
```

`namespace`

Type : chaîne

Obligatoire : non

Le paramètre du noyau de l'espace de noms pour lequel définir un `value` nom.

Valeurs d'espace de noms IPC valides : `"kernel.msgmax" | "kernel.msgmnb" | "kernel.msgmni" | "kernel.sem" | "kernel.shmall" | "kernel.shmmax" | "kernel.shmmni" | "kernel.shm_rmid_forced"`, et `Sysctls` qui commencent par `"fs.mqueue.*"`

Valeurs d'espace de noms de réseau valides : `Sysctls` qui commencent par `"net.*"`

`value`

Type : chaîne

Obligatoire : non

La valeur du paramètre de noyau de l'espace de noms spécifié dans`namespace`.

Interactive

`interactive`

Type : booléen

Obligatoire : non

Lorsque ce paramètre est `true`, vous pouvez déployer des applications conteneurisées qui nécessitent l'allocation d'une `stdin` ou d'un `tty`. Ce paramètre correspond `OpenStdin` à la commande docker create-container et à l'option `--interactive` docker run.

L'argument par défaut est `false`.

Pseudo Terminal

`pseudoTerminal`

Type : booléen

Obligatoire : non

Lorsque ce paramètre a la valeur `true`, un TTY est alloué. Ce paramètre correspond `Tty` à la commande docker create-container et à l'option `--tty` docker run.

L'argument par défaut est `false`.

Nom de l'accélérateur Elastic Inference

L'exigence de la ressource d'accélérateur Elastic Inference pour votre définition de tâche.

> **Note**
>
> Amazon Elastic Inference (EI) n'est plus disponible pour les clients.

Les paramètres suivants sont autorisés dans une définition de tâche :

`deviceName`

Type : String

Obligatoire : oui

Nom de l'appareil accélérateur d'inférence élastique pour l'instance. `deviceName` doit également être référencé dans une définition du conteneur. Consultez `Elastic Inference accelerator`.

`deviceType`

Type : String

Obligatoire : oui

L'accélérateur Elastic Inference à utiliser.

Contraintes de placement des tâches

Lorsque vous enregistrez une définition de tâche, vous pouvez fournir des contraintes de placement des tâches qui personnalisent la façon dont Amazon ECS place les tâches.

Vous pouvez utiliser des contraintes pour placer des tâches en fonction de la zone de disponibilité, du type d'instance ou d'attributs personnalisés. Pour de plus amples informations, veuillez consulter Définissez les instances de conteneur qu'Amazon ECS utilise pour les tâches.

Les paramètres suivants sont autorisés dans une définition de conteneur:

`expression`

Type : chaîne

Obligatoire : non

Expression de langage de requête de cluster à appliquer à la contrainte. Pour de plus amples informations, veuillez consulter Création d'expressions pour définir des instances de conteneur pour les tâches Amazon ECS.

`type`

Type : String

Obligatoire : oui

Type de contrainte. Utilisez `memberOf` pour limiter la sélection à un groupe de candidats valides.

Configuration du proxy

`proxyConfiguration`

Type : objet ProxyConfiguration

Obligatoire : non

Détails de configuration pour le proxy App Mesh.

Pour les tâches qui utilisent le type de EC2 lancement, les instances de conteneur nécessitent au moins la version 1.26.0 de l'agent de conteneur et au moins la version 1.26.0-1 du `ecs-init` package pour activer une configuration de proxy. Si vos instances de conteneur sont lancées à partir de la version d'AMI `20190301` ou ultérieure optimisée pour Amazon ECS, elles contiennent les versions requises de l'agent de conteneur et `ecs-init`. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "string",
    "properties": [
        {
            "name": "string",
```

```
            "value": "string"
        }
    ]
}
```

`type`

Type : String

Valeur valeurs : `APPMESH`

Obligatoire : non

Type de proxy. La seule valeur prise en charge est `APPMESH`.

`containerName`

Type : String

Obligatoire : oui

Nom du conteneur qui fait office de proxy App Mesh.

`properties`

Type : tableau d'objets KeyValuePair

Obligatoire : non

L'ensemble de paramètres de configuration réseau pour fournir le plug-in Container Network Interface (CNI), spécifié sous la forme de paires clé-valeur.

- `IgnoredUID` – (Obligatoire) ID d'utilisateur (UID) du conteneur de proxy, tel que défini par le paramètre `user` dans une définition de conteneur. Ce port est utilisé pour garantir que le proxy ignore son propre trafic. Si vous avez spécifié `IgnoredGID`, ce champ doit être vide.
- `IgnoredGID` – (Obligatoire) ID de groupe (GID) du conteneur de proxy, tel que défini par le paramètre `user` dans une définition de conteneur. Ce port est utilisé pour garantir que le proxy ignore son propre trafic. Si vous avez spécifié `IgnoredUID`, ce champ doit être vide.
- `AppPorts` - (Obligatoire) Liste des ports utilisés par l'application. Le trafic réseau vers ces ports est transmis à `ProxyIngressPort` et `ProxyEgressPort`.
- `ProxyIngressPort` – (Obligatoire) Spécifie le port auquel le trafic entrant vers `AppPorts` est dirigé.

- `ProxyEgressPort` – (Obligatoire) Spécifie le port auquel le trafic sortant de `AppPorts` est dirigé.
- `EgressIgnoredPorts` – (Obligatoire) Le trafic sortant accédant à ces ports spécifiés est ignoré et n'est pas redirigé vers le `ProxyEgressPort`. Cela peut être une liste vide.
- `EgressIgnoredIPs` – (Obligatoire) Le trafic sortant accédant à ces adresses IP spécifiées est ignoré et n'est pas redirigé vers le `ProxyEgressPort`. Cela peut être une liste vide.

`name`

Type : chaîne

Obligatoire : non

Nom de la paire clé-valeur.

`value`

Type : chaîne

Obligatoire : non

Valeur de la paire clé-valeur.

Volumes

Lorsque vous enregistrez une définition de tâche, vous pouvez éventuellement spécifier une liste de volumes à transmettre au Docker démon sur une instance de conteneur, qui sera ensuite accessible aux autres conteneurs de la même instance de conteneur.

Les types de volumes de données qui peuvent être utilisés sont les suivants :

- Volumes Amazon EBS : fournit un stockage par blocs rentable, durable et performant pour les charges de travail conteneurisées gourmandes en données. Vous pouvez joindre 1 volume Amazon EBS par tâche Amazon ECS lors de l'exécution d'une tâche autonome, ou lors de la création ou de la mise à jour d'un service. Les volumes Amazon EBS sont pris en charge pour les tâches Linux. Pour de plus amples informations, veuillez consulter Utiliser des volumes Amazon EBS avec Amazon ECS.
- Volumes Amazon EFS : fournit un stockage de fichiers simple, évolutif et permanent à utiliser avec vos tâches Amazon ECS. Avec Amazon EFS, la capacité de stockage est élastique. Elle augmente

et diminue automatiquement au fil de vos ajouts et suppressions de fichiers. Vos applications peuvent disposer de l'espace de stockage dont elles ont besoin, au moment où elles en ont besoin. Les volumes Amazon EFS sont pris en charge. Pour de plus amples informations, veuillez consulter Utiliser les volumes Amazon EFS avec Amazon ECS.

- FSx pour les volumes de serveurs de fichiers Windows : fournit des serveurs de fichiers Microsoft Windows entièrement gérés. Ces serveurs de fichiers sont basés sur un système de fichiers Windows. Lorsque vous utilisez FSx un serveur de fichiers Windows avec Amazon ECS, vous pouvez fournir à vos tâches Windows un stockage de fichiers persistant, distribué, partagé et statique. Pour de plus amples informations, veuillez consulter Utilisation FSx pour les volumes de serveurs de fichiers Windows avec Amazon ECS.

  Les conteneurs Windows sur Fargate ne prennent pas en charge cette option.
- Volumes Docker : volume géré par Docker créé sous `/var/lib/docker/volumes` l'instance Amazon hôte. EC2 Des pilotes de volume Docker (également appelés plug-ins) permettent d'intégrer les volumes avec des systèmes de stockage externe, tels qu'Amazon EBS. Le pilote de volume `local` intégré ou un pilote de volume tiers peut être utilisé. Les volumes Docker ne sont pris en charge que lors de l'exécution de tâches sur des EC2 instances Amazon. Les conteneurs Windows ne prennent en charge que l'utilisation du `local` pilote. Pour utiliser des volumes Docker, spécifiez un `dockerVolumeConfiguration` dans votre définition de tâche.
- Montages par liaison : fichier ou répertoire de la machine hôte monté dans un conteneur. Les volumes hôtes à montage par liaison sont pris en charge. Pour utiliser des volumes hôte de montage lié, spécifiez un `host` et éventuellement une valeur `sourcePath` dans votre définition de tâche.

Pour de plus amples informations, veuillez consulter Options de stockage pour les tâches Amazon ECS.

Les paramètres suivants sont autorisés dans une définition de conteneur.

`name`

Type : chaîne

Obligatoire : non

Nom du volume. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, tirets () et traits de soulignement (`-`) sont autorisés. _ Ce nom est référencé dans le `sourceVolume` paramètre de l'`mountPoints`objet de définition du conteneur.

`host`

Obligatoire : non

Le `host` paramètre est utilisé pour lier le cycle de vie du montage de liaison à l' EC2 instance Amazon hôte, plutôt qu'à la tâche et à l'endroit où il est stocké. Si le paramètre `host` est vide, le démon Docker attribue un chemin hôte au volume de données, mais la persistance des données après l'arrêt des conteneurs qui lui sont associés n'est pas garantie.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`.

> **Note**
>
> Le `sourcePath` paramètre n'est pris en charge que lors de l'utilisation de tâches hébergées sur EC2 des instances Amazon.

`sourcePath`

Type : chaîne

Obligatoire : non

Lorsque le `host` paramètre est utilisé, spécifiez a `sourcePath` pour déclarer le chemin sur l' EC2 instance Amazon hôte qui est présentée au conteneur. Si ce paramètre est vide, le démon Docker attribue un chemin hôte pour vous. Si le `host` paramètre contient un emplacement de `sourcePath` fichier, le volume de données est conservé à l'emplacement spécifié sur l' EC2 instance Amazon hôte jusqu'à ce que vous le supprimiez manuellement. Si la `sourcePath` valeur n'existe pas sur l' EC2 instance Amazon hôte, le démon Docker la crée. Si l'emplacement n'existe pas, le contenu du chemin source est exporté.

`configuredAtLaunch`

Type : booléen

Obligatoire : non

Spécifie si un volume est configurable au lancement. Lorsque ce paramètre est défini sur`true`, vous pouvez configurer le volume lors de l'exécution d'une tâche autonome ou lors de la création ou de la mise à jour d'un service. Lorsque ce paramètre est défini sur`true`, vous ne pourrez

pas fournir d'autre configuration de volume dans la définition de tâche. Ce paramètre doit être défini sur `true` pour configurer un volume Amazon EBS à associer à une tâche. Le paramétrage `configuredAtLaunch` `true` et le report de la configuration du volume jusqu'à la phase de lancement vous permettent de créer des définitions de tâches qui ne sont pas limitées à un type de volume ou à des paramètres de volume spécifiques. Cela rend votre définition de tâche réutilisable dans différents environnements d'exécution. Pour plus d'informations, consultez Volumes Amazon EBS.

`dockerVolumeConfiguration`

Type : DockerVolumeConfigurationObjet

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Docker. Les volumes Docker ne sont pris en charge que lors de l'exécution de tâches sur des EC2 instances. Les conteneurs Windows ne prennent en charge que l'utilisation du `local` pilote. Pour utiliser des montages liés, spécifiez plutôt un paramètre `host`.

`scope`

Type : String

Valeurs valides : `task` | `shared`

Obligatoire : non

Portée du volume Docker, qui détermine son cycle de vie. Les volumes Docker destinés à un élément `task` sont automatiquement mis en service lorsque la tâche commence, et détruits lorsque la tâche s'arrête. Les volumes Docker définis comme `shared` ne sont pas supprimés lorsque la tâche s'arrête.

`autoprovision`

Type : booléen

Valeur par défaut : `false`

Obligatoire : non

Si cette valeur est `true`, le volume Docker est créé s'il n'existe pas déjà. Ce champ n'est utilisé que si `scope` c'est le cas`shared`. Si tel `scope` est le cas`task`, ce paramètre doit être omis.

`driver`

Type : chaîne

Obligatoire : non

Pilote de volume Docker à utiliser. La valeur du pilote doit correspondre au nom du pilote fourni par Docker car ce nom est utilisé pour le placement des tâches. Si le pilote a été installé à l'aide de la CLI du plugin Docker, utilisez-le `docker plugin ls` pour récupérer le nom du pilote depuis votre instance de conteneur. Si le pilote a été installé à l'aide d'une autre méthode, utilisez Docker plugin discovery pour récupérer le nom du pilote.

`driverOpts`

Type : chaîne

Obligatoire : non

Une carte des options spécifiques au pilote Docker à utiliser. Ce paramètre correspond `DriverOpts` à la section Créer un volume de Docker.

`labels`

Type : chaîne

Obligatoire : non

Métadonnées personnalisées à ajouter à votre volume Docker.

`efsVolumeConfiguration`

Type : objet EFSVolumede configuration

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Amazon EFS.

`fileSystemId`

Type : String

Obligatoire : oui

ID du système de fichiers Amazon EFS à utiliser.

`rootDirectory`

Type : chaîne

Obligatoire : non

Répertoire du système de fichiers Amazon EFS à monter en tant que répertoire racine à l'intérieur de l'hôte. Si ce paramètre est omis, la racine du volume Amazon EFS est utilisée. La spécification de `/` a le même effet que l'omission de ce paramètre.

> **Important**
>
> Si un point d'accès EFS est spécifié dans le`authorizationConfig`, le paramètre du répertoire racine doit être omis ou défini sur`/`, ce qui appliquera le chemin défini sur le point d'accès EFS.

`transitEncryption`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Indique si vous souhaitez activer ou non le chiffrement des données Amazon EFS en transit entre l'hôte Amazon ECS et le serveur Amazon EFS. Si l'autorisation Amazon EFS IAM est utilisée, le chiffrement de transit doit être activé. Si ce paramètre est omis, la valeur par défaut `DISABLED` est utilisée. Pour plus d'informations, consultez Chiffrement des données en transit dans le Guide de l'utilisateur Amazon Elastic File System.

`transitEncryptionPort`

Type : entier

Obligatoire : non

Port à utiliser lors de l'envoi de données chiffrées entre l'hôte Amazon ECS et le serveur Amazon EFS. Si vous ne spécifiez pas de port de chiffrement de transit, la tâche utilisera la stratégie de sélection de port utilisée par l'assistant de montage Amazon EFS. Pour plus d'informations, consultez Assistant de montage EFS dans le Guide de l'utilisateur Amazon Elastic File System User.

`authorizationConfig`

Type : objet EFSAuthorizationConfig

Obligatoire : non

Détails de configuration des autorisations pour le système de fichiers Amazon EFS.

`accessPointId`

Type : chaîne

Obligatoire : non

ID du point d'accès à utiliser. Si un point d'accès est spécifié, la valeur du répertoire racine `efsVolumeConfiguration` doit être omise ou définie sur `/`, ce qui appliquera le chemin défini sur le point d'accès EFS. Si un point d'accès est utilisé, le chiffrement de transit doit être activé dans `EFSVolumeConfiguration`. Pour plus d'informations, consultez Utilisation des points d'accès Amazon EFS dans le Guide de l'utilisateur Amazon Elastic File System.

`iam`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Spécifie s'il faut utiliser le rôle IAM de tâche Amazon ECS défini dans une définition de tâche lors du montage du système de fichiers Amazon EFS. Si cette option est activée, le chiffrement en transit doit être activé dans la configuration `EFSVolumeConfiguration`. Si ce paramètre est omis, la valeur par défaut `DISABLED` est utilisée. Pour plus d'informations, consultez Rôles IAM pour les tâches.

`FSxWindowsFileServerVolumeConfiguration`

Type : FSxWindowsFileServerVolumeConfigurationObjet

Obligatoire : oui

Ce paramètre est spécifié lorsque vous utilisez un système de fichiers Amazon FSx pour Windows File Server pour le stockage des tâches.

`fileSystemId`

Type : String

Obligatoire : oui

L' FSx identifiant du système de fichiers Windows File Server à utiliser.

`rootDirectory`

Type : String

Obligatoire : oui

Le répertoire du système FSx de fichiers Windows File Server à monter en tant que répertoire racine sur l'hôte.

`authorizationConfig`

`credentialsParameter`

Type : String

Obligatoire : oui

Options d'informations d'identification d'autorisation.

Options :

- Amazon Resource Name (ARN) d'un AWS Secrets Managersecret.
- ARN d'un AWS Systems Managerparamètre.

`domain`

Type : String

Obligatoire : oui

Nom de domaine complet hébergé par un annuaire AWS Directory Service for Microsoft Active Directory(AWS Managed Microsoft AD) ou un Active Directory auto-hébergé EC2 .

Balises

Lorsque vous enregistrez une définition de tâche, vous pouvez éventuellement spécifier des étiquettes de métadonnées qui sont appliquées à la définition de tâche. Les balises vous aident à

classer et à organiser votre définition de tâche. Chaque balise est constituée d'une clé et d'une valeur facultative. Vous définissez ces deux éléments. Pour de plus amples informations, veuillez consulter Balisage des ressources Amazon ECS.

> **Important**
>
> N'ajoutez pas de données d'identification personnelle ou d'autres informations confidentielles ou sensibles dans les étiquettes. Les tags sont accessibles à de nombreux AWS services, y compris la facturation. Les étiquettes ne sont pas destinées à être utilisées pour des données privées ou sensibles.

Les paramètres suivants sont autorisés dans un objet balise.

`key`

Type : chaîne

Obligatoire : non

Partie d'une paire clé-valeur qui constitue une étiquette. Une clé est une étiquette générale qui fait office de catégorie pour les valeurs d'étiquette plus spécifiques.

`value`

Type : chaîne

Obligatoire : non

Partie facultative d'une paire clé-valeur qui constitue une étiquette. Une valeur agit comme un descripteur au sein d'une catégorie d'étiquette (clé).

Autres paramètres de définition de tâche

Les paramètres de définition de tâche suivants peuvent être utilisés lors de l'enregistrement des définitions de tâches dans la console Amazon ECS à l'aide de l'option Configure via JSON (Configurer via JSON). Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.

Rubriques

- Mode IPC
- Mode PID
- Injection de défauts

Mode IPC

`ipcMode`

Type : chaîne

Obligatoire : non

Espace de noms de ressource IPC à utiliser pour les conteneurs de la tâche. Les valeurs valides sont `host`, `task` ou `none`. Si cette option `host` est spécifiée, tous les conteneurs inclus dans les tâches qui ont spécifié le mode `host` IPC sur la même instance de conteneur partagent les mêmes ressources IPC avec l'instance Amazon EC2 hôte. Si `task` est spécifié, tous les conteneurs de la tâche spécifiée partagent les mêmes ressources IPC. Si `none` est spécifié, les ressources IPC des conteneurs d'une tâche sont privés et ne partagent rien avec les autres conteneurs d'une tâche ou d'une instance de conteneur. Si aucune valeur n'est spécifiée, le partage de l'espace de noms de ressource IPC dépend du paramètre du démon Docker sur l'instance de conteneur.

Si le mode IPC `host` est utilisé, il existe un risque accru d'exposition à un espace de noms IPC indésirable.

Si vous définissez les paramètres du noyau placés dans un espace de noms à l'aide de `systemControls` pour les conteneurs de la tâche, les éléments suivants s'appliquent à votre espace de noms de ressource IPC.

- Pour les tâches qui utilisent le mode IPC `host`, les `systemControls` liés à l'espace de noms IPC ne sont pas pris en charge.
- Pour les tâches qui utilisent le mode IPC `task`, les `systemControls` liés à l'espace de noms IPC s'appliquent à tous les conteneurs au sein d'une tâche.

Mode PID

`pidMode`

Type : String

Valeurs valides : `host` | `task`

Obligatoire : non

Espace de noms de processus à utiliser pour les conteneurs de la tâche. Les valeurs valides sont `host` ou `task`. Par exemple, la surveillance des sidecars peut avoir besoin de `pidMode` pour accéder à des informations sur d'autres conteneurs exécutés dans le cadre de la même tâche.

Si cette `host` option est spécifiée, tous les conteneurs des tâches qui ont spécifié le mode `host` PID sur la même instance de conteneur partagent le même espace de noms de processus avec l' EC2instance Amazon hôte.

Si `task` est spécifié, tous les conteneurs de la tâche spécifiée partagent le même espace de noms de processus.

Si aucune valeur n'est spécifiée, la valeur par défaut est un espace de noms privé pour chaque conteneur.

Si le mode PID `host` est utilisé, il existe un risque accru d'exposition à un espace de noms de processus indésirable.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

Injection de défauts

`enableFaultInjection`

Type : booléen

Valeurs valides : `true` | `false`

Obligatoire : non

Si ce paramètre est défini sur`true`, dans la charge utile d'une tâche, Amazon ECS accepte les demandes d'injection d'erreurs provenant des conteneurs de la tâche. Par défaut, ce paramètre est défini sur `false`.

Modèle de définition de tâche Amazon ECS

Un modèle de définition de tâche vide est présenté ci-dessous. Vous pouvez utiliser ce modèle pour créer votre définition de tâche, qui peut ensuite être collée dans la zone de saisie JSON de la console ou enregistrée dans un fichier et utilisée avec l' AWS CLI `--cli-input-jsonoption`. Pour de plus amples informations, veuillez consulter [Paramètres de définition des tâches Amazon ECS pour le type de lancement Fargate](#).

Modèle de type de EC2 lancement Amazon

```
{
  "family": "",
  "taskRoleArn": "",
  "executionRoleArn": "",
  "networkMode": "none",
  "containerDefinitions": [
    {
      "name": "",
      "image": "",
      "repositoryCredentials": {
        "credentialsParameter": ""
      },
      "cpu": 0,
      "memory": 0,
      "memoryReservation": 0,
      "links": [""],
      "portMappings": [
        {
          "containerPort": 0,
          "hostPort": 0,
          "protocol": "tcp"
        }
      ],
      "restartPolicy": {
        "enabled": true,
        "ignoredExitCodes": [0],
        "restartAttemptPeriod": 180
      },
      "essential": true,
      "entryPoint": [""],
      "command": [""],
      "environment": [
```

```
        {
          "name": "",
          "value": ""
        }
      ],
      "environmentFiles": [
        {
          "value": "",
          "type": "s3"
        }
      ],
      "mountPoints": [
        {
          "sourceVolume": "",
          "containerPath": "",
          "readOnly": true
        }
      ],
      "volumesFrom": [
        {
          "sourceContainer": "",
          "readOnly": true
        }
      ],
      "linuxParameters": {
        "capabilities": {
          "add": [""],
          "drop": [""]
        },
        "devices": [
          {
            "hostPath": "",
            "containerPath": "",
            "permissions": ["read"]
          }
        ],
        "initProcessEnabled": true,
        "sharedMemorySize": 0,
        "tmpfs": [
          {
            "containerPath": "",
            "size": 0,
            "mountOptions": [""]
          }
```

```
        ],
        "maxSwap": 0,
        "swappiness": 0
      },
      "secrets": [
        {
          "name": "",
          "valueFrom": ""
        }
      ],
      "dependsOn": [
        {
          "containerName": "",
          "condition": "COMPLETE"
        }
      ],
      "startTimeout": 0,
      "stopTimeout": 0,
      "hostname": "",
      "user": "",
      "workingDirectory": "",
      "disableNetworking": true,
      "privileged": true,
      "readonlyRootFilesystem": true,
      "dnsServers": [""],
      "dnsSearchDomains": [""],
      "extraHosts": [
        {
          "hostname": "",
          "ipAddress": ""
        }
      ],
      "dockerSecurityOptions": [""],
      "interactive": true,
      "pseudoTerminal": true,
      "dockerLabels": {
        "KeyName": ""
      },
      "ulimits": [
        {
          "name": "nofile",
          "softLimit": 0,
          "hardLimit": 0
        }
```

```
      ],
      "logConfiguration": {
        "logDriver": "splunk",
        "options": {
          "KeyName": ""
        },
        "secretOptions": [
          {
            "name": "",
            "valueFrom": ""
          }
        ]
      },
      "healthCheck": {
        "command": [""],
        "interval": 0,
        "timeout": 0,
        "retries": 0,
        "startPeriod": 0
      },
      "systemControls": [
        {
          "namespace": "",
          "value": ""
        }
      ],
      "resourceRequirements": [
        {
          "value": "",
          "type": "InferenceAccelerator"
        }
      ],
      "firelensConfiguration": {
        "type": "fluentbit",
        "options": {
          "KeyName": ""
        }
      }
    }
  ],
  "volumes": [
    {
      "name": "",
      "host": {
```

```
        "sourcePath": ""
      },
      "configuredAtLaunch": true,
      "dockerVolumeConfiguration": {
        "scope": "shared",
        "autoprovision": true,
        "driver": "",
        "driverOpts": {
          "KeyName": ""
        },
        "labels": {
          "KeyName": ""
        }
      },
      "efsVolumeConfiguration": {
        "fileSystemId": "",
        "rootDirectory": "",
        "transitEncryption": "DISABLED",
        "transitEncryptionPort": 0,
        "authorizationConfig": {
          "accessPointId": "",
          "iam": "ENABLED"
        }
      },
      "fsxWindowsFileServerVolumeConfiguration": {
        "fileSystemId": "",
        "rootDirectory": "",
        "authorizationConfig": {
          "credentialsParameter": "",
          "domain": ""
        }
      }
    }
  ],
  "placementConstraints": [
    {
      "type": "memberOf",
      "expression": ""
    }
  ],
  "requiresCompatibilities": ["EC2"],
  "cpu": "",
  "memory": "",
  "tags": [
```

```
    {
      "key": "",
      "value": ""
    }
  ],
  "pidMode": "task",
  "ipcMode": "task",
  "proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "",
    "properties": [
      {
        "name": "",
        "value": ""
      }
    ]
  },
  "inferenceAccelerators": [
    {
      "deviceName": "",
      "deviceType": ""
    }
  ],
  "ephemeralStorage": {
    "sizeInGiB": 0
  },
  "runtimePlatform": {
    "cpuArchitecture": "X86_64",
    "operatingSystemFamily": "WINDOWS_SERVER_20H2_CORE"
  }
}
```

Modèle de type de lancement Fargate

> **Important**
>
> Pour le type de lancement Fargate, vous devez inclure `operatingSystemFamily` le paramètre avec l'une des valeurs suivantes :
>
> - `LINUX`
> - `WINDOWS_SERVER_2019_FULL`

- WINDOWS_SERVER_2019_CORE
- WINDOWS_SERVER_2022_FULL
- WINDOWS_SERVER_2022_CORE

```
{
    "family": "",
    "runtimePlatform": {"operatingSystemFamily": ""},
    "taskRoleArn": "",
    "executionRoleArn": "",
    "networkMode": "awsvpc",
    "platformFamily": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            "repositoryCredentials": {"credentialsParameter": ""},
            "cpu": 0,
            "memory": 0,
            "memoryReservation": 0,
            "links": [""],
            "portMappings": [
                {
                    "containerPort": 0,
                    "hostPort": 0,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": [""],
            "command": [""],
            "environment": [
                {
                    "name": "",
                    "value": ""
                }
            ],
            "environmentFiles": [
                {
                    "value": "",
                    "type": "s3"
                }
```

```
            ],
            "mountPoints": [
                {
                    "sourceVolume": "",
                    "containerPath": "",
                    "readOnly": true
                }
            ],
            "volumesFrom": [
                {
                    "sourceContainer": "",
                    "readOnly": true
                }
            ],
            "linuxParameters": {
                "capabilities": {
                    "add": [""],
                    "drop": [""]
                },
                "devices": [
                    {
                        "hostPath": "",
                        "containerPath": "",
                        "permissions": ["read"]
                    }
                ],
                "initProcessEnabled": true,
                "sharedMemorySize": 0,
                "tmpfs": [
                    {
                        "containerPath": "",
                        "size": 0,
                        "mountOptions": [""]
                    }
                ],
                "maxSwap": 0,
                "swappiness": 0
            },
            "secrets": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ],
```

```
            "dependsOn": [
                {
                    "containerName": "",
                    "condition": "HEALTHY"
                }
            ],
            "startTimeout": 0,
            "stopTimeout": 0,
            "hostname": "",
            "user": "",
            "workingDirectory": "",
            "disableNetworking": true,
            "privileged": true,
            "readonlyRootFilesystem": true,
            "dnsServers": [""],
            "dnsSearchDomains": [""],
            "extraHosts": [
                {
                    "hostname": "",
                    "ipAddress": ""
                }
            ],
            "dockerSecurityOptions": [""],
            "interactive": true,
            "pseudoTerminal": true,
            "dockerLabels": {"KeyName": ""},
            "ulimits": [
                {
                    "name": "msgqueue",
                    "softLimit": 0,
                    "hardLimit": 0
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {"KeyName": ""},
                "secretOptions": [
                    {
                        "name": "",
                        "valueFrom": ""
                    }
                ]
            },
            "healthCheck": {
```

```
                "command": [""],
                "interval": 0,
                "timeout": 0,
                "retries": 0,
                "startPeriod": 0
            },
            "systemControls": [
                {
                    "namespace": "",
                    "value": ""
                }
            ],
            "resourceRequirements": [
                {
                    "value": "",
                    "type": "GPU"
                }
            ],
            "firelensConfiguration": {
                "type": "fluentd",
                "options": {"KeyName": ""}
            }
        }
    ],
    "volumes": [
        {
            "name": "",
            "host": {"sourcePath": ""},
            "configuredAtLaunch":true,
            "dockerVolumeConfiguration": {
                "scope": "task",
                "autoprovision": true,
                "driver": "",
                "driverOpts": {"KeyName": ""},
                "labels": {"KeyName": ""}
            },
            "efsVolumeConfiguration": {
                "fileSystemId": "",
                "rootDirectory": "",
                "transitEncryption": "ENABLED",
                "transitEncryptionPort": 0,
                "authorizationConfig": {
                    "accessPointId": "",
                    "iam": "ENABLED"
```

```
                }
            }
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "",
    "memory": "",
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "ephemeralStorage": {"sizeInGiB": 0},
    "pidMode": "task",
    "ipcMode": "none",
    "proxyConfiguration": {
        "type": "APPMESH",
        "containerName": "",
        "properties": [
            {
                "name": "",
                "value": ""
            }
        ]
    },
    "inferenceAccelerators": [
        {
            "deviceName": "",
            "deviceType": ""
        }
    ]
}
```

Vous pouvez générer ce modèle de définition de tâche à l'aide de la commande AWS CLI suivante.

```
aws ecs register-task-definition --generate-cli-skeleton
```

Exemples de définitions de tâches Amazon ECS

Vous pouvez copier les exemples et les extraits de code pour commencer à créer vos propres définitions de tâches.

Vous pouvez copier les exemples, puis les coller lorsque vous utilisez l'option Configurer via JSON dans la console classique. Assurez-vous de personnaliser les exemples, en utilisant par exemple votre ID de compte. Vous pouvez inclure les extraits dans le JSON de définition de tâche. Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console et Paramètres de définition des tâches Amazon ECS pour le type de lancement Fargate.

Pour d'autres exemples de définition de tâches, voir AWS Exemples de définitions de tâches sur GitHub.

Rubriques

- serveur Web
- splunkpilote de journal
- fluentdpilote de journal
- gelfpilote de journal
- Charges de travail sur les instances externes
- Image Amazon ECR et définition des tâches (rôle IAM)
- Point d'entrée avec commande
- Dépendances du conteneur
- Volumes dans les définitions de tâches
- Exemples de définitions de tâche Windows

serveur Web

Voici un exemple de définition de tâche à l'aide des conteneurs Linux du type de lancement Fargate qui configure un serveur web :

```
{
   "containerDefinitions": [
      {
         "command": [
            "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</
title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
         ],
```

```
        "entryPoint": [
           "sh",
           "-c"
        ],
        "essential": true,
        "image": "public.ecr.aws/docker/library/httpd:2.4",
        "logConfiguration": {
           "logDriver": "awslogs",
           "options": {
              "awslogs-group" : "/ecs/fargate-task-definition",
              "awslogs-region": "us-east-1",
              "awslogs-stream-prefix": "ecs"
           }
        },
        "name": "sample-fargate-app",
        "portMappings": [
           {
              "containerPort": 80,
              "hostPort": 80,
              "protocol": "tcp"
           }
        ]
      }
   ],
   "cpu": "256",
   "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
   "family": "fargate-task-definition",
   "memory": "512",
   "networkMode": "awsvpc",
   "runtimePlatform": {
        "operatingSystemFamily": "LINUX"
    },
   "requiresCompatibilities": [
       "FARGATE"
    ]
}
```

Voici un exemple de définition de tâche à l'aide des conteneurs Windows du type de lancement Fargate qui configure un serveur web :

```
{
    "containerDefinitions": [
        {
```

```
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

`splunk`pilote de journal

L'extrait suivant montre comment utiliser le pilote de journal `splunk` dans une définition de tâche qui envoie les journaux à un service distant. Le paramètre de jeton Splunk est spécifiée en tant qu'option secrète, car il peut être traité comme des données sensibles. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.

```
"containerDefinitions": [{
  "logConfiguration": {
```

```
   "logDriver": "splunk",
   "options": {
    "splunk-url": "https://cloud.splunk.com:8080",
    "tag": "tag_name",
   },
   "secretOptions": [{
    "name": "splunk-token",
    "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:splunk-token-
KnrBkD"
}],
```

`fluentd`pilote de journal

L'extrait suivant montre comment utiliser le pilote de journal `fluentd` dans une définition de tâche qui envoie les journaux à un service distant. La valeur `fluentd-address` est spécifiée en tant qu'option secrète, car elle peut être traitée comme des données sensibles. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.

```
"containerDefinitions": [{
 "logConfiguration": {
  "logDriver": "fluentd",
  "options": {
   "tag": "fluentd demo"
  },
  "secretOptions": [{
   "name": "fluentd-address",
   "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:fluentd-address-
KnrBkD"
  }]
 },
 "entryPoint": [],
 "portMappings": [{
             "hostPort": 80,
             "protocol": "tcp",
             "containerPort": 80
             },
             {
  "hostPort": 24224,
  "protocol": "tcp",
  "containerPort": 24224
 }]
}],
```

`gelf`pilote de journal

L'extrait suivant montre comment utiliser le pilote de journal `gelf` dans une définition de tâche qui envoie les journaux à un hôte distant exécutant Logstash qui accepte les journaux Gelf sous forme d'entrée. Pour de plus amples informations, veuillez consulter `logConfiguration`.

```
"containerDefinitions": [{
 "logConfiguration": {
  "logDriver": "gelf",
  "options": {
   "gelf-address": "udp://logstash-service-address:5000",
   "tag": "gelf task demo"
  }
 },
 "entryPoint": [],
 "portMappings": [{
   "hostPort": 5000,
   "protocol": "udp",
   "containerPort": 5000
  },
  {
   "hostPort": 5000,
   "protocol": "tcp",
   "containerPort": 5000
  }
 ]
}],
```

Charges de travail sur les instances externes

Lors de l'enregistrement d'une définition de tâche Amazon ECS, utilisez le paramètre `requiresCompatibilities` et spécifiez `EXTERNAL`, qui valide la compatibilité de la définition de tâche lors de l'exécution d'applications Amazon ECS sur vos instances externes. Si vous utilisez la console pour enregistrer une définition de tâche, vous devez utiliser l'éditeur JSON. Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.

> **Important**
>
> Si vos tâches nécessitent un rôle IAM d'exécution de tâche, assurez-vous qu'il est spécifié dans la définition de tâche.

Lorsque vous déployez votre application, utilisez le type de lancement `EXTERNAL` lors de la création de votre service ou de l'exécution de votre tâche autonome.

Voici un exemple de définition de tâche.

Linux

```
{
 "requiresCompatibilities": [
  "EXTERNAL"
 ],
 "containerDefinitions": [{
  "name": "nginx",
  "image": "public.ecr.aws/nginx/nginx:latest",
  "memory": 256,
  "cpu": 256,
  "essential": true,
  "portMappings": [{
   "containerPort": 80,
   "hostPort": 8080,
   "protocol": "tcp"
  }]
 }],
 "networkMode": "bridge",
 "family": "nginx"
}
```

Windows

```
{
 "requiresCompatibilities": [
  "EXTERNAL"
 ],
 "containerDefinitions": [{
  "name": "windows-container",
  "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-ltsc2019",
```

```
    "memory": 256,
    "cpu": 512,
    "essential": true,
    "portMappings": [{
     "containerPort": 80,
     "hostPort": 8080,
     "protocol": "tcp"
    }]
   }],
   "networkMode": "bridge",
   "family": "windows-container"
  }
```

Image Amazon ECR et définition des tâches (rôle IAM)

L'extrait suivant utilise une image Amazon ECR appelée `aws-nodejs-sample` avec la balise `v1` à partir du registre `123456789012.dkr.ecr.us-west-2.amazonaws.com`. Le conteneur de cette tâche hérite des autorisations IAM du rôle `arn:aws:iam::123456789012:role/AmazonECSTaskS3BucketRole`. Pour de plus amples informations, veuillez consulter Rôle IAM de la tâche Amazon ECS.

```
{
    "containerDefinitions": [
        {
            "name": "sample-app",
            "image": "123456789012.dkr.ecr.us-west-2.amazonaws.com/aws-nodejs-
sample:v1",
            "memory": 200,
            "cpu": 10,
            "essential": true
        }
    ],
    "family": "example_task_3",
    "taskRoleArn": "arn:aws:iam::123456789012:role/AmazonECSTaskS3BucketRole"
}
```

Point d'entrée avec commande

L'extrait suivant illustre la syntaxe d'un conteneur Docker qui utilise un point d'entrée et un argument de commande. Ce conteneur enverra un ping à `example.com` quatre fois, puis il s'arrêtera.

```
{
    "containerDefinitions": [
        {
            "memory": 32,
            "essential": true,
            "entryPoint": ["ping"],
            "name": "alpine_ping",
            "readonlyRootFilesystem": true,
            "image": "alpine:3.4",
            "command": [
                "-c",
                "4",
                "example.com"
            ],
            "cpu": 16
        }
    ],
    "family": "example_task_2"
}
```

Dépendances du conteneur

Cet extrait illustre la syntaxe d'une définition de tâche avec plusieurs conteneurs où la dépendance de conteneur est spécifiée. Dans la définition de tâche suivante, le conteneur `envoy` doit atteindre un état sain, déterminé par les paramètres de surveillance de l'état du conteneur requis avant que le conteneur `app` démarre. Pour de plus amples informations, veuillez consulter Dépendances du conteneur.

```
{
  "family": "appmesh-gateway",
  "runtimePlatform": {
        "operatingSystemFamily": "LINUX"
  },
  "proxyConfiguration":{
      "type": "APPMESH",
      "containerName": "envoy",
      "properties": [
          {
              "name": "IgnoredUID",
              "value": "1337"
          },
          {
```

```
                "name": "ProxyIngressPort",
                "value": "15000"
            },
            {
                "name": "ProxyEgressPort",
                "value": "15001"
            },
            {
                "name": "AppPorts",
                "value": "9080"
            },
            {
                "name": "EgressIgnoredIPs",
                "value": "169.254.170.2,169.254.169.254"
            }
        ]
    },
    "containerDefinitions": [
      {
        "name": "app",
        "image": "application_image",
        "portMappings": [
          {
            "containerPort": 9080,
            "hostPort": 9080,
            "protocol": "tcp"
          }
        ],
        "essential": true,
        "dependsOn": [
          {
            "containerName": "envoy",
            "condition": "HEALTHY"
          }
        ]
      },
      {
        "name": "envoy",
        "image": "840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-
envoy:v1.15.1.0-prod",
        "essential": true,
        "environment": [
          {
            "name": "APPMESH_VIRTUAL_NODE_NAME",
```

```
          "value": "mesh/meshName/virtualNode/virtualNodeName"
        },
        {
          "name": "ENVOY_LOG_LEVEL",
          "value": "info"
        }
      ],
      "healthCheck": {
        "command": [
          "CMD-SHELL",
          "echo hello"
        ],
        "interval": 5,
        "timeout": 2,
        "retries": 3
      }
    }
  ],
  "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
  "networkMode": "awsvpc"
}
```

Volumes dans les définitions de tâches

Utilisez ce qui suit pour comprendre comment spécifier des volumes dans les tâches.

- Pour plus d'informations sur la configuration d'un volume Amazon EBS, consultezSpécifiez la configuration du volume Amazon EBS lors du déploiement d'Amazon ECS.
- Pour plus d'informations sur la configuration d'un volume Amazon EFS, consultezConfiguration des systèmes de fichiers Amazon EFS pour Amazon ECS à l'aide de la console.
- Pour plus d'informations sur la configuration d'un volume FSx pour un serveur de fichiers Windows, consultezDécouvrez comment configurer FSx les systèmes de fichiers Windows File Server pour Amazon ECS.
- Pour plus d'informations sur la configuration d'un volume Docker, consultezExemples de volumes Docker pour Amazon ECS.
- Pour plus d'informations sur la configuration d'un montage par liaison, consultezExemples de montage par liaison pour Amazon ECS.

Exemples de définitions de tâche Windows

Voici un exemple de définition de tâche qui peut vous aider à commencer à utiliser les conteneurs Windows sur Amazon ECS.

Example Exemple d'application de console Amazon ECS pour Windows

La définition de tâche suivante est l'exemple d'application de console Amazon ECS produit dans l'assistant de première exécution pour Amazon ECS ; il a été transféré pour utiliser l'image de conteneur `microsoft/iis` Windows.

```
{
  "family": "windows-simple-iis",
  "containerDefinitions": [
    {
      "name": "windows_sample_app",
      "image": "mcr.microsoft.com/windows/servercore/iis",
      "cpu": 1024,
      "entryPoint":["powershell", "-Command"],
      "command":["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file -
Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
      "portMappings": [
        {
          "protocol": "tcp",
          "containerPort": 80
        }
      ],
      "memory": 1024,
      "essential": true
    }
  ],
  "networkMode": "awsvpc",
  "memory": "1024",
  "cpu": "1024"
}
```

Clusters Amazon ECS

Un cluster Amazon ECS est un regroupement logique de tâches ou de services. Outre les tâches et les services, un cluster comprend les ressources suivantes :

- La capacité de l'infrastructure, qui peut être une combinaison des éléments suivants :
  - EC2 Instances Amazon dans le AWS cloud
  - Sans serveur (AWS Fargate) dans le cloud AWS
  - Machines virtuelles (VM) ou serveurs sur site
- Réseau (VPC et sous-réseau) sur lequel s'exécutent vos tâches et services

  Lorsque vous utilisez des EC2 instances Amazon pour la capacité, le sous-réseau peut se trouver dans des zones de disponibilité, des zones locales, des zones de longueur d'onde ou AWS Outposts.
- Un espace de noms facultatif

  L'espace de noms est utilisé pour service-to-service la communication avec Service Connect.
- Option de surveillance

  CloudWatch Container Insights est un service entièrement géré, moyennant un coût supplémentaire. Il collecte, agrège et résume automatiquement les métriques et les journaux Amazon ECS.

Vous trouverez ci-après des concepts généraux sur les clusters Amazon ECS.

- Vous créez des clusters pour séparer vos ressources.
- Les clusters sont Région AWS spécifiques.
- Les clusters peuvent présenter l'un des états suivants.

  ACTIF

  Le cluster est prêt à accepter des tâches et, le cas échéant, vous pouvez enregistrer des instances de conteneur auprès du cluster.

  PROVISIONING

  Le cluster est associé à des fournisseurs de capacité et les ressources nécessaires au fournisseur de capacité sont en cours de création.

DEPROVISIONING

Le cluster est associé à des fournisseurs de capacité et les ressources nécessaires au fournisseur de capacité sont en cours de suppression.

ÉCHEC

Le cluster est associé à des fournisseurs de capacité et la création des ressources nécessaires au fournisseur de capacité a échoué.

INACTIVE

Le cluster a été supprimé. Les clusters ayant un état `INACTIVE` peuvent rester détectables dans votre compte pendant un certain temps. Ce comportement est susceptible de changer à l'avenir. Veillez donc à ne pas vous fier à la persistance `INACTIVE` des clusters.

- Vous pouvez utiliser différents types d'instances pour le type de EC2 lancement ou les fournisseurs de capacité du groupe Auto Scaling. Une instance ne peut être enregistrée que dans un seul cluster à la fois.
- Vous pouvez restreindre l'accès aux clusters en créant des politiques IAM personnalisées. Pour plus d'informations, voir Exemples de clusters Amazon ECS la section dansExemples de politiques basées sur l'identité pour Amazon Elastic Container Service.
- Vous pouvez utiliser Service Auto Scaling pour dimensionner les tâches Fargate. Pour de plus amples informations, veuillez consulter Faites évoluer automatiquement votre service Amazon ECS.
- Vous pouvez configurer un espace de noms Service Connect par défaut pour un cluster. Une fois que vous avez défini un espace de noms Service Connect par défaut, tous les nouveaux services créés dans le cluster peuvent être ajoutés en tant que services clients dans l'espace de noms en activant Service Connect. Aucune configuration supplémentaire n’est requise. Pour de plus amples informations, veuillez consulter Utilisez Service Connect pour connecter les services Amazon ECS avec des noms abrégés.

Fournisseurs de capacité

Les fournisseurs de capacité Amazon ECS gèrent la mise à l'échelle de l'infrastructure pour les tâches de vos clusters. Chaque cluster dispose d'un ou plusieurs fournisseurs de capacité et éventuellement d'une stratégie de fournisseur de capacité facultative. Vous pouvez attribuer une stratégie de fournisseur de capacité par défaut au cluster. La stratégie de fournisseur de capacité détermine la façon dont les tâches sont réparties entre les fournisseurs de capacité du cluster.

Lorsque vous exécutez une tâche autonome ou créez un service, vous utilisez soit la stratégie de fournisseur de capacité par défaut du cluster, soit une stratégie de fournisseur de capacité qui remplace celle par défaut. La stratégie de fournisseur de capacité par défaut du cluster ne s'applique que lorsque vous ne spécifiez pas de type de lancement ou de stratégie de fournisseur de capacité pour votre tâche ou service. Si vous fournissez l'un de ces paramètres, la stratégie par défaut n'est pas utilisée.

Pour Fargate, il n'est pas nécessaire de créer ou de gérer la capacité. Il vous suffit d'associer au cluster l'un des fournisseurs de capacité prédéfinis suivants :

- Fargate
- Fargate Spot

Lorsque vous utilisez EC2 des instances pour votre capacité, vous utilisez le groupe Auto Scaling pour gérer les EC2 instances. Auto Scaling permet de s'assurer que vous disposez du nombre correct d' EC2 instances disponibles pour gérer le chargement de l'application.

Un cluster peut contenir une combinaison de tâches hébergées sur AWS Fargate des EC2 instances Amazon ou des instances externes. Les tâches peuvent être exécutées sur Fargate EC2 ou sur l'infrastructure en tant que type de lancement ou stratégie de fournisseur de capacités. Si vous l'utilisez EC2 comme type de lancement, Amazon ECS ne suit ni ne fait évoluer la capacité des groupes Amazon EC2 Auto Scaling. Pour plus d'informations sur les types de lancement, consultez Types de lancement Amazon ECS.

Un cluster peut contenir une combinaison à la fois de fournisseurs de capacité de groupe Auto Scaling et de fournisseurs de capacité Fargate. Une stratégie de fournisseur de capacité ne peut inclure que des fournisseurs de capacité du groupe Auto Scaling ou des fournisseurs de capacité Fargate.

Clusters Amazon ECS pour Fargate

Les fournisseurs de capacité Amazon ECS gèrent la mise à l'échelle de l'infrastructure pour les tâches de vos clusters. Chaque cluster dispose d'un ou plusieurs fournisseurs de capacité et éventuellement d'une stratégie de fournisseur de capacité facultative. La stratégie de fournisseur de capacité détermine la façon dont les tâches sont réparties entre les fournisseurs de capacité du cluster. Lorsque vous exécutez une tâche autonome ou que vous créez un service, vous pouvez utiliser soit la stratégie de fournisseur de capacité par défaut du cluster, soit une stratégie de fournisseur de capacité qui remplace celle par défaut.

Lorsque vous exécutez vos tâches AWS Fargate, vous n'avez pas besoin de créer ou de gérer la capacité. Il vous suffit d'associer au cluster l'un des fournisseurs de capacité prédéfinis suivants :

- Fargate
- Fargate Spot

Avec Amazon ECS sur les fournisseurs AWS Fargate de capacité, vous pouvez utiliser à la fois les capacités Fargate et Fargate Spot pour vos tâches Amazon ECS.

Avec Fargate Spot, vous pouvez exécuter des tâches Amazon ECS tolérantes aux interruptions à un tarif réduit par rapport au prix de Fargate. Fargate Spot exécute les tâches sur la capacité de calcul de réserve. Lorsque la AWS capacité est rétablie, vos tâches sont interrompues par un avertissement de deux minutes.

Lorsque les tâches qui utilisent les fournisseurs de capacité Fargate et Fargate Spot sont arrêtées, l'événement de changement d'état de la tâche est envoyé à Amazon. EventBridge Le motif de l'arrêt en décrit la cause. Pour de plus amples informations, veuillez consulter Événements de modification de l'état des tâches Amazon ECS.

Un cluster peut contenir une combinaison à la fois de fournisseurs de capacité Fargate et de groupe Auto Scaling. Toutefois, une stratégie de fournisseur de capacité ne peut contenir que des fournisseurs de capacité Fargate ou de groupe Auto Scaling, mais pas les deux. Pour plus d'informations, consultez Auto Scaling Group Capacity Providers.

Prenez les points suivants en compte lors de l'utilisation de fournisseurs de capacité :

- Vous devez associer un fournisseur de capacité à un cluster avant de l'associer à la stratégie du fournisseur de capacité.
- Vous pouvez spécifier un maximum de 20 fournisseurs de capacité pour une stratégie de fournisseur de capacité.
- Vous ne pouvez pas mettre à jour un service utilisant un fournisseur de capacité de groupe Auto Scaling pour utiliser un fournisseur de capacité Fargate. L'inverse est également vrai.
- Dans une stratégie de fournisseur de capacité, si aucune valeur `weight` n'est spécifiée pour un fournisseur de capacité dans la console, alors la valeur par défaut `1` est utilisée. Si vous utilisez l'API ou AWS CLI, la valeur par défaut de `0` est utilisée.
- Lorsque plusieurs fournisseurs de capacité sont spécifiés dans le cadre d'une stratégie de fournisseur de capacité, au moins l'un des fournisseurs de capacité doit disposer d'une valeur de poids supérieure à zéro. Les fournisseurs de capacité dont le coefficient de pondération est

nul ne sont pas habitués à attribuer des tâches. Si vous spécifiez, dans une stratégie, plusieurs fournisseurs de capacité qui possèdent tous un poids nul, toutes les actions `RunTask` ou `CreateService` utilisant la stratégie de fournisseur de capacité échoueront.

- Une valeur de base ne peut être définie que pour un seul fournisseur de capacité dans une stratégie de fournisseur de capacité. Si aucune valeur de base n'est spécifiée, la valeur par défaut de zéro est utilisée.
- Un cluster peut contenir une combinaison à la fois de fournisseurs de capacité de groupe Auto Scaling et de fournisseurs de capacité Fargate. Toutefois, une stratégie de fournisseur de capacité ne peut contenir que des fournisseurs de capacité du groupe Auto Scaling ou Fargate, mais pas les deux.
- Un cluster peut contenir une combinaison de services et de tâches autonomes utilisant à la fois des fournisseurs de capacité et des types de lancement. Un service peut être mis à jour pour utiliser une stratégie de fournisseur de capacité plutôt qu'un type de lancement. Vous devez toutefois forcer un nouveau déploiement pour ce faire.

Avis de résiliation de Fargate Spot

Pendant les périodes de forte demande, la capacité Spot de Fargate peut être indisponible. Cela peut retarder les tâches Spot de Fargate. Dans ce cas, les services Amazon ECS réessayent de lancer des tâches jusqu'à ce que la capacité requise soit disponible. Fargate ne remplace pas la capacité Spot par une capacité à la demande.

Lorsque des tâches utilisant la capacité Fargate Spot sont arrêtées en raison d'une interruption Spot, un avertissement d'interruption sous deux minutes est envoyé avant cet arrêt. L'avertissement est envoyé en tant qu'événement de changement d'état de tâche à Amazon EventBridge et en tant que signal SIGTERM à la tâche en cours d'exécution. Si vous utilisez Fargate Spot au titre d'un service, le planificateur de service reçoit dans ce scénario le signal d'interruption et tente de lancer des tâches supplémentaires sur Fargate Spot si la capacité est disponible. Un service avec une seule tâche est interrompu jusqu'à ce que la capacité soit disponible. Pour plus d'informations sur un arrêt progressif, veuillez consulter le billet de blog Graceful shutdowns with ECS.

Pour faire en sorte que vos conteneurs se ferment correctement avant l'arrêt de la tâche, vous pouvez configurer les éléments suivants :

- Il est possible de spécifier une valeur de temporisation d'arrêt (`stopTimeout`) de `120` secondes ou moins dans la définition de conteneur utilisée par la tâche. La valeur `stopTimeout` par défaut est de 30 secondes. Vous pouvez spécifier une valeur `stopTimeout` plus longue pour disposer

de plus de temps entre le moment où l'événement de changement d'état de tâche est reçu et l'instant où le conteneur est arrêté de force. Pour de plus amples informations, veuillez consulter Temporisations de conteneurs.

- Le signal SIGTERM doit être reçu à l'intérieur du conteneur pour effectuer des actions de nettoyage. Si vous ne parvenez pas à traiter ce signal, la tâche reçoit un signal SIGKILL après le `stopTimeout` configuré et peut entraîner une perte ou une corruption de données.

Voici un extrait d'événement de changement d'état de tâche. Cet extrait indique le motif et le code d'arrêt d'une interruption Fargate Spot.

```
{
  "version": "0",
  "id": "9bcdac79-b31f-4d3d-9410-fbd727c29fab",
  "detail-type": "ECS Task State Change",
  "source": "aws.ecs",
  "account": "111122223333",
  "resources": [
    "arn:aws:ecs:us-east-1:111122223333:task/b99d40b3-5176-4f71-9a52-9dbd6f1cebef"
  ],
  "detail": {
    "clusterArn": "arn:aws:ecs:us-east-1:111122223333:cluster/default",
    "createdAt": "2016-12-06T16:41:05.702Z",
    "desiredStatus": "STOPPED",
    "lastStatus": "RUNNING",
    "stoppedReason": "Your Spot Task was interrupted.",
    "stopCode": "SpotInterruption",
    "taskArn": "arn:aws:ecs:us-east-1:111122223333:task/
b99d40b3-5176-4f71-9a52-9dbd6fEXAMPLE",
    ...
  }
}
```

Voici un modèle d'événement utilisé pour créer une EventBridge règle pour les événements de changement d'état des tâches Amazon ECS. Vous pouvez éventuellement spécifier un cluster dans le champ `detail`. Cela signifie que vous recevrez des événements de changement d'état de tâche pour ce cluster. Pour plus d'informations sur la création d'une EventBridge règle, consultez Getting started with Amazon EventBridge dans le guide de EventBridge l'utilisateur Amazon.

```
{
    "source": [
```

```
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Task State Change"
    ],
    "detail": {
        "clusterArn": [
            "arn:aws:ecs:us-west-2:111122223333:cluster/default"
        ]
    }
}
```

Création d'un cluster Amazon ECS pour le type de lancement Fargate

Vous créez un cluster pour définir l'infrastructure sur laquelle s'exécutent vos tâches et services.

Avant de commencer, veillez achever les étapes de Configurer l'utilisation d'Amazon ECS et affectez l'autorisation IAM appropriée. Pour de plus amples informations, veuillez consulter the section called "Exemples de clusters Amazon ECS". La console Amazon ECS crée les ressources nécessaires à un cluster Amazon ECS en créant une AWS CloudFormation pile.

La console associe automatiquement les fournisseurs de capacité Fargate et Fargate Spot au cluster.

Vous pouvez modifier les options suivantes :

- Ajoutez un espace de noms au cluster.

  Un espace de noms permet aux services que vous créez dans le cluster de se connecter aux autres services de l'espace de noms sans configuration supplémentaire. Pour de plus amples informations, veuillez consulter Interconnectez les services Amazon ECS.
- Activez Container Insights avec une observabilité améliorée, ou Container Insights.

  CloudWatch Container Insights collecte, agrège et résume les métriques et les journaux de vos applications conteneurisées et de vos microservices. Conteneur Insights fournit également des informations de diagnostic (par exemple sur les échecs de redémarrage des conteneurs) que vous pouvez utiliser pour isoler les problèmes et les résoudre rapidement. Pour de plus amples informations, veuillez consulter the section called "Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée".

  Le 2 décembre 2024, AWS a publié Container Insights avec une observabilité améliorée pour Amazon ECS. Cette version prend en charge une observabilité améliorée pour les clusters Amazon

ECS utilisant les types de lancement Amazon EC2 et Fargate. Une fois que vous avez configuré Container Insights avec une observabilité améliorée sur Amazon ECS, Container Insights collecte automatiquement des données télémétriques détaillées sur l'infrastructure, du niveau du cluster au niveau du conteneur dans votre environnement, et affiche vos données dans des tableaux de bord qui vous présentent diverses mesures et dimensions. Vous pouvez ensuite utiliser ces out-of-the-box tableaux de bord sur la console Container Insights pour mieux comprendre l'état et les performances de vos conteneurs, et pour atténuer les problèmes plus rapidement en identifiant les anomalies.

Nous vous recommandons d'utiliser Container Insights avec une observabilité améliorée plutôt que Container Insights, car cela fournit une visibilité détaillée de votre environnement de conteneurs, réduisant ainsi le délai moyen de résolution.

- Ajoutez des balises pour vous aider à identifier vos clusters.
- Attribuez une AWS KMS clé à votre espace de stockage géré. Pour plus d'informations sur la création d'une clé, voir Création d'une clé KMS dans le Guide de AWS Key Management Service l'utilisateur.
- Attribuez une AWS KMS clé à votre espace de stockage éphémère Fargate. Pour plus d'informations sur la création d'une clé, voir Création d'une clé KMS dans le Guide de AWS Key Management Service l'utilisateur.

Procédure

Pour créer un nouveau cluster (console Amazon ECS)

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Configuration de cluster, configurez les éléments suivants :

   - Pour Nom du cluster, saisissez un nom unique.

     Le nom peut contenir jusqu'à 255 lettres (minuscules et majuscules), des chiffres et des traits d'union.
   - (Facultatif) Pour que l'espace de noms utilisé pour Service Connect soit différent du nom du cluster, saisissez un nom unique dans Espace de nom.

6. (Facultatif) Utilisez Container Insights, développez Monitoring, puis choisissez l'une des options suivantes :

   - Pour utiliser le Container Insights recommandé avec une observabilité améliorée, choisissez Container Insights avec une observabilité améliorée.
   - Pour utiliser Container Insights, choisissez Container Insights.

7. (Facultatif). Sous Chiffrement, vous pouvez configurer les éléments suivants :

   - Chiffrez vos données sur le stockage éphémère Fargate. Sous Chiffrement, pour le stockage éphémère Fargate, entrez l'ARN de la clé que vous souhaitez utiliser pour chiffrer AWS KMS les données de stockage éphémère Fargate.
   - Chiffrez les données sur le stockage géré. Sous Chiffrement, pour Stockage géré, entrez l'ARN de la AWS KMS clé que vous souhaitez utiliser pour chiffrer les données du stockage géré.

8. (Facultatif) Pour vous aider à identifier votre cluster, développez Tags (balises), puis configurez vos balises.

   [Add a tag] Choisissez Add tag (Ajouter une balise) et procédez comme suit :

   - Pour Key (Clé), saisissez le nom de la clé.
   - Pour Value (Valeur), saisissez la valeur de clé.

   [Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de l'étiquette.

9. Choisissez Créer.

Étapes suivantes

Après avoir créé le cluster, vous pouvez créer des définitions de tâches pour vos applications, puis les exécuter en tant que tâches autonomes ou dans le cadre d'un service. Pour plus d’informations, consultez les ressources suivantes :

- Définitions de tâche Amazon ECS
- Exécution d'une application en tant que tâche Amazon ECS
- Création d'un déploiement de mises à jour continues Amazon ECS

Fournisseurs de capacité Amazon ECS pour le type de EC2 lancement

Lorsque vous utilisez EC2 des instances Amazon pour votre capacité, vous utilisez les groupes Auto Scaling pour gérer les EC2 instances Amazon enregistrées dans leurs clusters. Auto Scaling permet de s'assurer que vous disposez du nombre correct d' EC2 instances Amazon disponibles pour gérer le chargement de l'application.

Vous pouvez utiliser la fonctionnalité de dimensionnement géré pour qu'Amazon ECS gère les actions de scale-in et de scale-out du groupe Auto Scaling, ou vous pouvez gérer les actions de dimensionnement vous-même. Pour de plus amples informations, veuillez consulter Gérez automatiquement la capacité d'Amazon ECS grâce au dimensionnement automatique des clusters.

Nous vous recommandons de créer un nouveau groupe Auto Scaling vide. Si vous utilisez un groupe Auto Scaling existant, les EC2 instances Amazon associées au groupe qui étaient déjà en cours d'exécution et enregistrées dans un cluster Amazon ECS avant que le groupe Auto Scaling ne soit utilisé pour créer un fournisseur de capacité ne soient pas correctement enregistrées auprès du fournisseur de capacité. Cela peut occasionner des problèmes lorsque le fournisseur de capacité est utilisé dans une stratégie de fournisseur de capacité. Utilisez `DescribeContainerInstances` pour vérifier qu'une instance de conteneur est bien associée à un fournisseur de capacité.

> **Note**
>
> Pour créer un groupe Auto Scaling vide, définissez le nombre souhaité sur zéro. Après avoir créé le fournisseur de capacité et l'avoir associé à un cluster, vous pouvez le mettre à l'échelle.
> Lorsque vous utilisez la console Amazon ECS, Amazon ECS crée un modèle de EC2 lancement Amazon et un groupe Auto Scaling en votre nom dans le cadre de la AWS CloudFormation pile. Ils sont préfixés par`EC2ContainerService-<ClusterName>`. Vous pouvez utiliser le groupe Auto Scaling comme fournisseur de capacité pour ce cluster.

Nous vous recommandons d'utiliser le drainage d'instance géré pour permettre la résiliation progressive des EC2 instances Amazon sans perturber vos charges de travail. Cette fonctionnalité est activée par défaut. Pour de plus amples informations, consultez Arrêtez en toute sécurité les charges de travail Amazon ECS exécutées sur les instances EC2 .

Les points suivants doivent être pris en compte lors de l'utilisation de fournisseurs de capacité de groupe Auto Scaling dans la console :

- La valeur du paramètre `MaxSize` d'un groupe Auto Scaling doit être supérieure à zéro pour une montée en puissance.
- Le groupe Auto Scaling ne peut pas avoir de paramètres de pondération d'instance.
- Si le groupe Auto Scaling ne peut pas monter en puissance pour s'adapter au nombre de tâches exécutées, les tâches ne parviennent pas à passer outre l'état `PROVISIONING`.
- Ne modifiez pas la ressource de politique de mise à l'échelle associée à vos groupes Auto Scaling gérés par des fournisseurs de capacité.
- Si la mise à l'échelle gérée est activée au moment où vous créez un fournisseur de capacité, le nombre souhaité de groupes Auto Scaling peut être défini sur `0`. Lorsque la mise à l'échelle gérée est activée, Amazon ECS gère les actions de mise à l'échelle horizontale et de montée en puissance du groupe Auto Scaling.
- Vous devez associer le fournisseur de capacité à un cluster avant de l'associer à la stratégie du fournisseur de capacité.
- Vous pouvez spécifier un maximum de 20 fournisseurs de capacité pour une stratégie de fournisseur de capacité.
- Vous ne pouvez pas mettre à jour un service utilisant un fournisseur de capacité de groupe Auto Scaling pour utiliser un fournisseur de capacité Fargate. L'inverse est également vrai.
- Dans une stratégie de fournisseur de capacité, si aucune valeur `weight` n'est spécifiée pour un fournisseur de capacité dans la console, alors la valeur par défaut `1` est utilisée. Si vous utilisez l'API ou AWS CLI, la valeur par défaut de `0` est utilisée.
- Lorsque plusieurs fournisseurs de capacité sont spécifiés dans le cadre d'une stratégie de fournisseur de capacité, au moins l'un des fournisseurs de capacité doit disposer d'une valeur de poids supérieure à zéro. Les fournisseurs de capacité dont le coefficient de pondération est nul ne sont pas habitués à attribuer des tâches. Si vous spécifiez, dans une stratégie, plusieurs fournisseurs de capacité qui possèdent tous un poids nul, toutes les actions `RunTask` ou `CreateService` utilisant la stratégie de fournisseur de capacité échoueront.
- Une valeur de base ne peut être définie que pour un seul fournisseur de capacité dans une stratégie de fournisseur de capacité. Si aucune valeur de base n'est spécifiée, la valeur par défaut de zéro est utilisée.
- Un cluster peut contenir une combinaison à la fois de fournisseurs de capacité de groupe Auto Scaling et de fournisseurs de capacité Fargate. Toutefois, une stratégie de fournisseur de capacité

ne peut contenir que des fournisseurs de capacité du groupe Auto Scaling ou Fargate, mais pas les deux.

- Un cluster peut contenir une combinaison de services et de tâches autonomes utilisant à la fois des fournisseurs de capacité et des types de lancement. Un service peut être mis à jour pour utiliser une stratégie de fournisseur de capacité plutôt qu'un type de lancement. Vous devez toutefois forcer un nouveau déploiement pour ce faire.
- Amazon ECS prend en charge les pools chauds Amazon EC2 Auto Scaling. Un warm pool est un groupe d' EC2 instances Amazon pré-initialisées prêtes à être mises en service. Chaque fois que votre application doit évoluer, Amazon EC2 Auto Scaling utilise les instances préinitialisées issues du warm pool plutôt que de lancer des instances froides. Cela permet à tout processus d'initialisation final de s'exécuter avant que l'instance ne soit mise en service. Pour de plus amples informations, veuillez consulter Configuration d'instances pré-initialisées pour votre groupe Amazon ECS Auto Scaling.

Pour plus d'informations sur la création d'un modèle de lancement Amazon EC2 Auto Scaling, consultez la section Modèles de lancement Auto Scaling dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling. Pour plus d'informations sur la création d'un groupe Amazon EC2 Auto Scaling, consultez la section Groupes Auto Scaling dans le guide de l'utilisateur Amazon EC2 Auto Scaling.

Considérations relatives à la sécurité des instances de EC2 conteneur Amazon pour Amazon ECS

Vous devez envisager une instance de conteneur unique et son accès dans le cadre de votre modèle de menace. Par exemple, une seule tâche affectée peut être en mesure de tirer parti des autorisations IAM d'une tâche non infectée sur la même instance.

Nous vous recommandons d'utiliser la procédure suivante pour empêcher cela :

- N'utilisez pas les privilèges d'administrateur lors de l'exécution de vos tâches.
- Attribuez un rôle de tâche avec un accès de moindre privilège à vos tâches.

  L'agent de conteneur crée automatiquement un jeton avec un ID d'informations d'identification unique qui est utilisé pour accéder aux ressources Amazon ECS.
- Pour empêcher les conteneurs exécutés par des tâches utilisant le mode `awsvpc` réseau d'accéder aux informations d'identification fournies au profil d' EC2 instance Amazon, tout en autorisant les autorisations fournies par le rôle de tâche, définissez la variable de configuration

de l'`ECS_AWSVPC_BLOCK_IMDS`agent sur true dans le fichier de configuration de l'agent et redémarrez l'agent.

- Utilisez Amazon GuardDuty Runtime Monitoring pour détecter les menaces visant les clusters et les conteneurs au sein de votre AWS environnement. La surveillance du temps d'exécution utilise un agent de GuardDuty sécurité qui augmente la visibilité de l'exécution sur les charges de travail Amazon ECS individuelles, par exemple l'accès aux fichiers, l'exécution des processus et les connexions réseau. Pour plus d'informations, consultez la section Surveillance du temps GuardDuty d'exécution dans le guide de GuardDuty l'utilisateur.

Création d'un cluster Amazon ECS pour le type de EC2 lancement Amazon

Vous créez un cluster pour définir l'infrastructure sur laquelle s'exécutent vos tâches et services.

Avant de commencer, veillez achever les étapes de Configurer l'utilisation d'Amazon ECS et affectez l'autorisation IAM appropriée. Pour de plus amples informations, veuillez consulter the section called “Exemples de clusters Amazon ECS”. La console Amazon ECS fournit un moyen simple de créer les ressources nécessaires à un cluster Amazon ECS en créant une AWS CloudFormation pile.

Pour que le processus de création de cluster soit aussi simple que possible, la console dispose de sélections par défaut pour de nombreux choix que nous décrivons ci-dessous. La plupart des sections de la console comportent des volets d'aide qui fournissent un contexte supplémentaire.

Vous pouvez enregistrer EC2 des instances Amazon lorsque vous créez le cluster ou enregistrer des instances supplémentaires auprès du cluster une fois celui-ci créé.

Vous pouvez modifier les options par défaut suivantes :

- Modifiez les sous-réseaux sur lesquels vos instances sont lancées.
- Modifiez les groupes de sécurité utilisés pour contrôler le trafic vers les instances de conteneur.
- Ajoutez un espace de noms au cluster.

  Un espace de noms permet aux services que vous créez dans le cluster de se connecter aux autres services de l'espace de noms sans configuration supplémentaire. Pour de plus amples informations, veuillez consulter Interconnectez les services Amazon ECS.
- Activez Container Insights avec une observabilité améliorée, ou Container Insights.

  CloudWatch Container Insights collecte, agrège et résume les métriques et les journaux de vos applications conteneurisées et de vos microservices. Conteneur Insights fournit également des

informations de diagnostic (par exemple sur les échecs de redémarrage des conteneurs) que vous pouvez utiliser pour isoler les problèmes et les résoudre rapidement. Pour de plus amples informations, veuillez consulter the section called “Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée”.

Le 2 décembre 2024, AWS a publié Container Insights avec une observabilité améliorée pour Amazon ECS. Cette version prend en charge une observabilité améliorée pour les clusters Amazon ECS utilisant les types de lancement Amazon EC2 et Fargate. Une fois que vous avez configuré Container Insights avec une observabilité améliorée sur Amazon ECS, Container Insights collecte automatiquement des données télémétriques détaillées sur l'infrastructure, du niveau du cluster au niveau du conteneur dans votre environnement, et affiche vos données dans des tableaux de bord qui vous présentent diverses mesures et dimensions. Vous pouvez ensuite utiliser ces out-of-the-box tableaux de bord sur la console Container Insights pour mieux comprendre l'état et les performances de vos conteneurs, et pour atténuer les problèmes plus rapidement en identifiant les anomalies.

Nous vous recommandons d'utiliser Container Insights avec une observabilité améliorée plutôt que Container Insights, car cela fournit une visibilité détaillée de votre environnement de conteneurs, réduisant ainsi le délai moyen de résolution.

- Attribuez une AWS KMS clé à votre espace de stockage géré. Pour plus d'informations sur la création d'une clé, voir Création d'une clé KMS dans le Guide de AWS Key Management Service l'utilisateur.
- Attribuez une AWS KMS clé à votre espace de stockage éphémère Fargate. Pour plus d'informations sur la création d'une clé, voir Création d'une clé KMS dans le Guide de AWS Key Management Service l'utilisateur.
- Ajoutez des balises pour vous aider à identifier vos clusters.

Options de groupe Auto Scaling

Lorsque vous utilisez des EC2 instances Amazon, vous devez spécifier un groupe Auto Scaling pour gérer l'infrastructure sur laquelle vos tâches et services s'exécutent.

Lorsque vous choisissez de créer un groupe Auto Scaling, il est automatiquement configuré pour le comportement suivant :

- Amazon ECS gère les actions de mise à l'échelle horizontale et de montée en puissance du groupe Auto Scaling.

- Amazon ECS n'empêchera pas la mise hors service des EC2 instances Amazon contenant des tâches et appartenant à un groupe Auto Scaling lors d'une action de scale-in. Pour plus d'informations, consultez Protection des instances dans le Guide de l'utilisateur AWS Auto Scaling .

Vous configurez les propriétés de groupe Auto Scaling suivantes qui déterminent le type et le nombre d'instances à lancer pour le groupe :

- L'AMI optimisée pour Amazon ECS.
- Type d'instance.
- La paire de clés SSH qui prouve votre identité lorsque vous connectez à l'instance. Pour plus d'informations sur la création de clés SSH, consultez les paires de EC2 clés Amazon et les instances Linux dans le guide de l' EC2 utilisateur Amazon.
- Nombre minimum d'instances à lancer pour le groupe Auto Scaling.
- Nombre maximal d'instances démarrées pour le groupe Auto Scaling.

  Pour que le groupe puisse monter en puissance, le maximum doit être supérieur à 0.

Amazon ECS crée un modèle de lancement Amazon EC2 Auto Scaling et un groupe Auto Scaling en votre nom dans le cadre de la AWS CloudFormation pile. Les valeurs que vous avez spécifiées pour l'AMI, les types d'instances et la paire de clés SSH se trouvent dans le modèle de lancement. Les modèles sont précédés du préfixe `EC2ContainerService-<ClusterName>`, ce qui facilite leur identification. Les groupes Auto Scaling sont précédés du préfixe `<ClusterName>-ECS-Infra-ECSAutoScalingGroup`.

Les instances lancées pour le groupe Auto Scaling utilisent le modèle de lancement.

Options de mise en réseau

Par défaut, les instances sont lancées dans les sous-réseaux par défaut de la région. Les groupes de sécurité, qui contrôlent le trafic vers vos instances de conteneur, actuellement associés aux sous-réseaux sont utilisés. Vous pouvez modifier les sous-réseaux et les groupes de sécurité des instances.

Vous pouvez choisir un sous-réseau existant. Vous pouvez soit utiliser un groupe de sécurité existant, soit en créer un nouveau. Lorsque vous créez un nouveau groupe de sécurité, vous devez spécifier au moins une règle entrante.

Les règles entrantes déterminent le trafic qui peut atteindre vos instances de conteneur et incluent les propriétés suivantes :

- Le protocole à autoriser
- La plage de ports à autoriser
- Le trafic entrant (source)

Pour autoriser le trafic entrant provenant d'une adresse ou d'un bloc d'adresses CIDR spécifique, utilisez Personnalisée pour Source avec le CIDR autorisé.

Pour autoriser le trafic entrant en provenance de toutes les destinations, utilisez Partout pour Source. Cela ajoute automatiquement le bloc d'adresse CIDR 0.0.0.0/0 et le bloc d' IPv4 adresse CIDR : :/0. IPv6

Pour autoriser le trafic entrant depuis votre ordinateur local, utilisez Groupe source pour Source. Cette action ajoute automatiquement l'adresse IP actuelle de votre ordinateur local comme source autorisée.

Pour créer un nouveau cluster (console Amazon ECS)

Avant de commencer, attribuez l'autorisation IAM adéquate. Pour de plus amples informations, veuillez consulter the section called “Exemples de clusters Amazon ECS”.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Configuration de cluster, configurez les éléments suivants :

   - Pour Nom du cluster, saisissez un nom unique.

     Le nom peut contenir jusqu'à 255 lettres (minuscules et majuscules), des chiffres et des traits d'union.
   - (Facultatif) Pour que l'espace de noms utilisé pour Service Connect soit différent du nom du cluster, saisissez un nom unique dans Espace de nom.
6. Ajoutez des EC2 instances Amazon à votre cluster, étendez l'infrastructure, puis sélectionnez EC2 les instances Amazon.

Ensuite, configurez le groupe Auto Scaling qui agit en tant que fournisseur de capacité :

a. Pour utiliser un groupe Auto Scaling existant, depuis Auto Scaling group (ASG) (Groupe Auto Scaling [ASG]), sélectionnez le groupe.

b. Pour créer un groupe Auto Scaling, depuis Auto Scaling group (ASG) (Groupe Auto Scaling [ASG]), sélectionnez Create new group (Créer un nouveau groupe), puis fournissez les détails suivants sur le groupe :

   - Pour le modèle de provisionnement, choisissez d'utiliser des instances à la demande ou des instances ponctuelles.
   - Si vous choisissez d'utiliser des instances ponctuelles, dans le cadre de la stratégie d'allocation, choisissez les pools de capacité ponctuels (types d'instances et zones de disponibilité) utilisés pour les instances.

     Pour la plupart des charges de travail, vous pouvez choisir Price capacity optimized.

     Pour plus d'informations, consultez la section Stratégies d'allocation pour les instances Spot dans le guide de EC2 l'utilisateur Amazon.
   - Pour l'instance de conteneur Amazon Machine Image (AMI), choisissez l'AMI optimisée pour Amazon ECS pour les instances du groupe Auto Scaling.
   - EC2 Par type d'instance, choisissez le type d'instance pour vos charges de travail.

     La mise à l'échelle gérée fonctionne mieux si votre groupe Auto Scaling utilise les mêmes types d'instance ou des types d'instance similaires.
   - Par EC2 exemple, choisissez un rôle d'instance de conteneur existant ou créez-en un nouveau.

     Pour de plus amples informations, veuillez consulter Rôle IAM d'instance de conteneur Amazon ECS.
   - Pour Capacity (Capacité), saisissez le nombre minimum et le nombre maximum d'instances à lancer dans le groupe Auto Scaling.
   - Pour SSH key pair (Paire de clés SSH), choisissez la paire qui prouve votre identité lorsque vous connectez à l'instance.
   - Pour permettre une image et un stockage plus grands, pour la taille du volume Root EBS, entrez la valeur en GiB.

7. (Facultatif) Pour modifier le VPC et les sous-réseaux, sous Mise en réseau pour les EC2 instances Amazon, effectuez l'une des opérations suivantes :

   - Pour supprimer un sous-réseau, sous Subnets (Sous-réseaux), choisissez X pour chaque sous-réseau que vous souhaitez supprimer.
   - Pour passer à un VPC autre que celui par défaut, sous VPC, choisissez un VPC existant, puis sous Sous-réseaux, sélectionnez les sous-réseaux.
   - Choisissez les groupes de sécurité. Sous Groupe de sécurité, choisissez l'une des options suivantes :
     - Pour utiliser un groupe de sécurité existant, choisissez Utiliser un groupe de sécurité existant, puis sélectionnez le groupe de sécurité.
     - Pour créer un groupe de sécurité, sélectionnez Créer un nouveau groupe de sécurité. Ensuite, choisissez Ajouter une règle pour chaque règle entrante.

       Pour plus d'informations sur les règles entrantes, veuillez consulter Options de mise en réseau.
   - Pour attribuer automatiquement des adresses IP publiques à vos instances de EC2 conteneur Amazon, dans Attribuer automatiquement une adresse IP publique, choisissez l'une des options suivantes :
     - Utiliser le paramètre de sous-réseau : attribuez une adresse IP publique aux instances lorsque le sous-réseau dans lequel les instances sont lancées est un sous-réseau public.
     - Activer : attribuez une adresse IP publique aux instances.
8. (Facultatif) Utilisez Container Insights, développez Monitoring, puis choisissez l'une des options suivantes :

   - Pour utiliser le Container Insights recommandé avec une observabilité améliorée, choisissez Container Insights avec une observabilité améliorée.
   - Pour utiliser Container Insights, choisissez Container Insights.
9. (Facultatif)

   Si vous utilisez la surveillance du temps d'exécution avec l'option manuelle et que vous souhaitez que ce cluster soit surveillé GuardDuty, choisissez Ajouter une balise et procédez comme suit :

   - Pour Key, entrez **`guardDutyRuntimeMonitoringManaged`**
   - Pour le champ Valeur, saisissez **`true`**.

10. (Facultatif) Chiffrez les données sur le stockage géré. Sous Chiffrement, pour Stockage géré, entrez l'ARN de la AWS KMS clé que vous souhaitez utiliser pour chiffrer les données du stockage géré.
11. (Facultatif) Pour gérer les identifications de cluster, développez Tags (Identifications), puis effectuez l'une des opérations suivantes :

    [Add a tag] Choisissez Add tag (Ajouter une étiquette) et procédez comme suit :

    - Pour Key (Clé), saisissez le nom de la clé.
    - Pour Value (Valeur), saisissez la valeur de clé.

    [Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de l'étiquette.
12. Choisissez Créer.

Étapes suivantes

Après avoir créé le cluster, vous pouvez créer des définitions de tâches pour vos applications, puis les exécuter en tant que tâches autonomes ou dans le cadre d'un service. Pour plus d'informations, consultez les ressources suivantes :

- Définitions de tâche Amazon ECS
- Exécution d'une application en tant que tâche Amazon ECS
- Création d'un déploiement de mises à jour continues Amazon ECS

Gérez automatiquement la capacité d'Amazon ECS grâce au dimensionnement automatique des clusters

Amazon ECS peut gérer le dimensionnement des EC2 instances Amazon enregistrées dans votre cluster. C'est ce qu'on appelle l'autoscaling du cluster Amazon ECS. Vous activez le dimensionnement géré lorsque vous créez le fournisseur de capacité du groupe Amazon ECS Auto Scaling. Ensuite, vous définissez un pourcentage cible (le`targetCapacity`) pour l'utilisation de l'instance dans ce groupe Auto Scaling. Amazon ECS crée deux CloudWatch métriques personnalisées et une politique de dimensionnement du suivi des cibles pour votre groupe Auto Scaling. Amazon ECS gère ensuite les actions de scale-in et de scale-out en fonction de l'utilisation des ressources utilisées par vos tâches.

Pour chaque fournisseur de capacité de groupe Auto Scaling associé à un cluster, Amazon ECS crée et gère les ressources suivantes :

- Une CloudWatch alarme de faible valeur métrique
- Une CloudWatch alarme à valeur métrique élevée
- Stratégie de mise à l'échelle de suivi cible

> **Note**
>
> Amazon ECS crée la politique de mise à l'échelle de suivi cible et la joint au groupe Auto Scaling. Pour mettre à jour la stratégie de dimensionnement de suivi cible, mettez à jour les paramètres de dimensionnement géré du fournisseur de capacité au lieu de mettre à jour directement la politique de dimensionnement.

Lorsque vous désactivez le dimensionnement géré ou que vous dissociez le fournisseur de capacité d'un cluster, Amazon ECS supprime à la fois les CloudWatch métriques et les ressources de politique de dimensionnement de suivi des cibles.

Amazon ECS utilise les mesures suivantes pour déterminer les mesures à prendre :

`CapacityProviderReservation`

Pourcentage d'instances de conteneur utilisées par un fournisseur de capacité spécifique. Amazon ECS génère cette métrique.

Amazon ECS définit la valeur `CapacityProviderReservation` sur un nombre compris entre 0 et 100. Amazon ECS utilise la formule suivante pour représenter le ratio de capacité restant dans le groupe Auto Scaling. Amazon ECS publie ensuite la métrique sur CloudWatch. Pour plus d'informations sur le mode de calcul de la métrique, consultez Deep Dive on Amazon ECS Cluster Auto Scaling.

```
CapacityProviderReservation = (number of instances needed) / (number of running
 instances) x 100
```

`DesiredCapacity`

Quantité de capacité pour le groupe Auto Scaling. Cette métrique n'est pas publiée sur CloudWatch.

Amazon ECS publie la `CapacityProviderReservation` métrique CloudWatch dans l'espace de `AWS/ECS/ManagedScaling` noms. La métrique `CapacityProviderReservation` entraîne l'une des actions suivantes :

La valeur **`CapacityProviderReservation`** est égale à **`targetCapacity`**.

Le groupe Auto Scaling n'a pas besoin d'une mise à l'échelle horizontale ou d'une montée en puissance. Le pourcentage d'utilisation cible a été atteint.

La valeur **`CapacityProviderReservation`** est supérieure à **`targetCapacity`**.

Un plus grand nombre de tâches utilisent un pourcentage de capacité supérieur à votre pourcentage `targetCapacity`. L'augmentation de la valeur de la `CapacityProviderReservation` métrique entraîne l'action de l' CloudWatch alarme associée. Cette alarme met à jour la valeur `DesiredCapacity` pour le groupe Auto Scaling. Le groupe Auto Scaling utilise cette valeur pour lancer EC2 des instances, puis les enregistrer auprès du cluster.

Lorsque la valeur par défaut de `targetCapacity` est de 100 %, les nouvelles tâches sont à l'état `PENDING` pendant la montée en puissance, car les instances ne disposent pas de la capacité nécessaire pour exécuter les tâches. Une fois les nouvelles instances enregistrées auprès d'ECS, ces tâches commenceront sur les nouvelles instances.

La valeur **`CapacityProviderReservation`** est inférieure à **`targetCapacity`**.

Un plus petit nombre de tâches utilisent un pourcentage de capacité inférieur à votre pourcentage `targetCapacity`, et au moins une instance peut être résiliée. La diminution de la valeur de la `CapacityProviderReservation` métrique entraîne l'action de l' CloudWatch alarme associée. Cette alarme met à jour la valeur `DesiredCapacity` pour le groupe Auto Scaling. Le groupe Auto Scaling utilise cette valeur pour mettre fin aux instances de EC2 conteneur, puis les désenregistrer du cluster.

Le groupe Auto Scaling suit les stratégies de résiliation du groupe pour déterminer quelles instances seront résiliées en premier lors d'événements de mise à l'échelle horizontale. De plus, cela évite les instances ayant un paramètre de protection contre la mise à l'échelle horizontale d'instance activé. L'autoscaling de cluster permet de gérer les instances dotées du paramètre de protection contre la mise à l'échelle horizontale des instances si vous activez la protection contre la résiliation gérée. Pour plus d'informations sur la protection contre la résiliation gérée, veuillez consulter Contrôlez les instances auxquelles Amazon ECS met fin. Pour plus d'informations sur la manière dont les groupes Auto Scaling mettent fin aux instances, consultez la section Contrôler

quelles instances Auto Scaling mettent fin au scalage dans le manuel Amazon EC2 Auto Scaling User Guide.

Les points suivants doivent être pris en compte lors de l'utilisation de l'autoscaling de cluster :

- Ne modifiez pas ou ne gérez pas la capacité souhaitée pour le groupe Auto Scaling associé à un fournisseur de capacité avec d'autres stratégies de mise à l'échelle que celle gérée par Amazon ECS.
- Lorsqu'Amazon ECS passe à zéro instance, il lance automatiquement 2 instances.
- Amazon ECS utilise le rôle IAM `AWSServiceRoleForECS` lié à un service pour obtenir les autorisations dont il a besoin pour appeler en votre AWS Auto Scaling nom. Pour de plus amples informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS.
- Lorsque vous utilisez des fournisseurs de capacité avec des groupes Auto Scaling, l'utilisateur, le groupe ou le rôle qui crée les fournisseurs de capacité a besoin de l'autorisation `autoscaling:CreateOrUpdateTags`. En effet, Amazon ECS ajoute une balise au groupe Auto Scaling lorsqu'il l'associe au fournisseur de capacité.

  > **Important**
  >
  > Assurez-vous que les outils que vous utilisez ne suppriment pas la balise `AmazonECSManaged` du groupe Auto Scaling. Si cette balise est supprimée, Amazon ECS ne peut pas gérer le dimensionnement.

- Le dimensionnement automatique du cluster ne modifie pas le MinimumCapacityou MaximumCapacitypour le groupe. Pour que le groupe puisse être redimensionné, la valeur de MaximumCapacitydoit être supérieure à zéro.
- Lorsque la fonction Auto Scaling (mise à l'échelle gérée) est activée, un fournisseur de capacité ne peut être connecté qu'à un seul cluster à la fois. Si votre fournisseur de capacité a désactivé la mise à l'échelle gérée, vous pouvez l'associer à plusieurs clusters.
- Lorsque la mise à l'échelle gérée est désactivée, le fournisseur de capacité ne bénéficie pas d'une mise à l'échelle horizontale ou d'une montée en puissance. Vous pouvez utiliser une stratégie de fournisseur de capacité pour équilibrer vos tâches entre les fournisseurs de capacité.
- La `binpack` stratégie est la plus efficace en termes de capacité.
- Lorsque la capacité cible est inférieure à 100 %, la stratégie de placement doit être utilisée ; la `binpack` stratégie doit avoir un ordre supérieur à la `spread` stratégie. Cela empêche le

fournisseur de capacité de s'étendre jusqu'à ce que chaque tâche dispose d'une instance dédiée ou que la limite soit atteinte.

Activation de l'autoscaling de cluster

Vous pouvez activer le dimensionnement automatique du cluster à l'aide de la console ou du AWS CLI.

Lorsque vous créez un cluster pour le type de EC2 lancement à l'aide de la console, Amazon ECS crée un groupe Auto Scaling en votre nom et définit la capacité cible. Pour de plus amples informations, veuillez consulter Création d'un cluster Amazon ECS pour le type de EC2 lancement Amazon.

Vous pouvez également créer un groupe Auto Scaling, puis l'attribuer à un cluster. Pour de plus amples informations, veuillez consulter Mettre à jour un fournisseur de capacité Amazon ECS.

Lorsque vous utilisez le AWS CLI, après avoir créé le cluster

1. Avant de créer le fournisseur de capacité, vous devez créer un groupe Auto Scaling. Pour plus d'informations, consultez la section Groupes Auto Scaling dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling.
2. `put-cluster-capacity-providers`À utiliser pour modifier le fournisseur de capacité du cluster. Pour de plus amples informations, veuillez consulter Activation de la mise à l'échelle automatique du cluster Amazon ECS.

Optimisation de la mise à l'échelle automatique du cluster Amazon ECS

Les clients qui exécutent Amazon ECS sur Amazon EC2 peuvent tirer parti du dimensionnement automatique des clusters pour gérer le dimensionnement des groupes Amazon EC2 Auto Scaling. Grâce à la mise à l'échelle automatique du cluster, vous pouvez configurer Amazon ECS pour dimensionner automatiquement votre groupe Auto Scaling et vous concentrer uniquement sur l'exécution de vos tâches. Amazon ECS garantit que le groupe Auto Scaling évolue en fonction des besoins, sans qu'aucune autre intervention ne soit requise. Les fournisseurs de capacité Amazon ECS sont utilisés pour gérer l'infrastructure de votre cluster en veillant à ce qu'il y ait suffisamment d'instances de conteneur pour répondre aux demandes de votre application. Pour découvrir comment fonctionne la mise à l'échelle automatique des clusters, consultez Deep Dive on Amazon ECS Cluster Auto Scaling.

Le dimensionnement automatique du cluster repose sur une intégration CloudWatch basée sur le groupe Auto Scaling pour ajuster la capacité du cluster. Par conséquent, il a une latence inhérente associée à

- Publier les CloudWatch statistiques,
- Le temps nécessaire à la métrique `CapacityProviderReservation` pour violer les CloudWatch alarmes (haute et faible)
- Le temps nécessaire à l'échauffement d'une EC2 instance Amazon récemment lancée. Vous pouvez prendre les mesures suivantes pour rendre le dimensionnement automatique des clusters plus réactif afin d'accélérer les déploiements :

Dimensionnement par étapes du fournisseur de capacités

Les fournisseurs de capacité Amazon ECS grow/shrink fourniront les instances de conteneur répondant aux exigences de votre application. Le nombre minimum d'instances qu'Amazon ECS lancera est fixé à 1 par défaut. Cela peut prolonger la durée de vos déploiements, si plusieurs instances sont nécessaires pour placer vos tâches en attente. Vous pouvez augmenter le `minimumScalingStepSize`via l'API Amazon ECS afin d'augmenter le nombre minimum d'instances qu'Amazon ECS fait évoluer ou non à la fois. Une `maximumScalingStepSize`valeur trop faible peut limiter le nombre d'instances de conteneur augmentées ou dédimensionnées à la fois, ce qui peut ralentir vos déploiements.

> **Note**
>
> Cette configuration n'est actuellement disponible que via le `CreateCapacityProvider`ou `UpdateCapacityProvider` APIs.

Période de préchauffage de l'instance

La période de préchauffage de l'instance est la période au bout de laquelle une EC2 instance Amazon récemment lancée peut contribuer aux CloudWatch statistiques du groupe Auto Scaling. Une fois la période de préchauffage spécifiée expirée, l'instance est prise en compte dans les mesures agrégées du groupe Auto Scaling, et le dimensionnement automatique du cluster poursuit sa prochaine itération de calculs pour estimer le nombre d'instances requises.

La valeur par défaut `instanceWarmupPeriod`est de 300 secondes, que vous pouvez configurer à une valeur inférieure via `CreateCapacityProvider`ou `UpdateCapacityProvider` APIs pour

une mise à l'échelle plus réactive. Nous vous recommandons de définir la valeur sur une valeur supérieure à 60 secondes afin d'éviter le surprovisionnement.

Capacité de réserve

Si votre fournisseur de capacité ne dispose d'aucune instance de conteneur pour placer des tâches, il doit augmenter (augmenter) la capacité du cluster en lançant des EC2 instances Amazon à la volée et en attendant qu'elles démarrent avant de pouvoir y lancer des conteneurs. Cela peut réduire considérablement le taux de lancement des tâches. Deux options s'offrent à vous.

Dans ce cas, le fait de disposer d'une EC2 capacité Amazon disponible déjà lancée et prête à exécuter des tâches augmentera le taux de lancement effectif des tâches. Vous pouvez utiliser la `Target Capacity` configuration pour indiquer que vous souhaitez conserver de la capacité inutilisée dans vos clusters. Par exemple, en définissant `Target Capacity` 80 %, vous indiquez que votre cluster a besoin de 20 % de capacité de réserve à tout moment. Cette capacité inutilisée peut permettre le lancement immédiat de toutes les tâches autonomes, garantissant ainsi que les lancements de tâches ne sont pas ralentis. L'inconvénient de cette approche est l'augmentation potentielle des coûts liés au maintien de la capacité inutilisée des clusters.

Une autre approche que vous pouvez envisager consiste à augmenter la marge de manœuvre de votre service, et non celle du fournisseur de capacité. Cela signifie qu'au lieu de réduire la `Target Capacity` configuration pour libérer de la capacité inutilisée, vous pouvez augmenter le nombre de répliques dans votre service en modifiant la métrique de dimensionnement du suivi des cibles ou les seuils de dimensionnement par étapes du dimensionnement automatique du service. Notez que cette approche ne sera utile que pour les charges de travail exigeantes, mais n'aura aucun effet lorsque vous déployez de nouveaux services et que vous passez de 0 à N tâches pour la première fois. Pour plus d'informations sur les politiques de dimensionnement associées, consultez Target Tracking Scaling Policies ou Step Scaling Policies dans le manuel Amazon Elastic Container Service Developer Guide.

Comportement de dimensionnement géré par Amazon ECS

Lorsque vous avez des fournisseurs de capacité de groupe Auto Scaling qui utilisent le dimensionnement géré, Amazon ECS estime le nombre optimal d'instances à ajouter à votre cluster et utilise cette valeur pour déterminer le nombre d'instances à demander ou à libérer.

Comportement de montée en puissance gérée

Amazon ECS sélectionne un fournisseur de capacité pour chaque tâche en suivant la stratégie du fournisseur de capacité définie dans le service, la tâche autonome ou la valeur par défaut du cluster. Amazon ECS suit le reste de ces étapes pour un seul fournisseur de capacité.

Les tâches sans stratégie de fournisseur de capacité sont ignorées par les fournisseurs de capacités. Une tâche en attente qui n'est pas associée à une stratégie de fournisseur de capacité n'entraînera pas la montée en puissance d'un fournisseur de capacité. Les tâches ou les services ne peuvent pas définir de stratégie de fournisseur de capacité si elles définissent un type de lancement.

Le comportement de montée en puissance est décrit plus en détail ci-dessous.

- Regroupez toutes les tâches de provisionnement pour ce fournisseur de capacité de façon à ce chaque groupe ait exactement les mêmes besoins en ressources.
- Lorsque vous utilisez plusieurs types d'instances dans un groupe Auto Scaling, les types d'instances du groupe Auto Scaling sont triés par leurs paramètres. Ces paramètres incluent le vCPU, la mémoire, les interfaces réseau élastiques (ENIs), les ports et. GPUs Les types d'instance les plus petits et les plus grands pour chaque paramètre sont sélectionnés. Pour plus d'informations sur le choix du type d'instance, consultezInstances de EC2 conteneur Amazon pour Amazon ECS.

  > **Important**
  >
  > Si les besoins en ressources d'un groupe de tâches sont supérieurs à ceux du plus petit type d'instance du groupe Auto Scaling, ce groupe de tâches ne peut pas être exécuté avec ce fournisseur de capacité. Le fournisseur de capacité ne met pas le groupe Auto Scaling à l'échelle. Les tâches restent à l'état `PROVISIONING`.
  > Pour éviter que les tâches ne restent à l'état `PROVISIONING`, nous vous recommandons de créer des groupes Auto Scaling et des fournisseurs de capacité distincts pour des besoins en ressources minimales différents. Lorsque vous exécutez des tâches ou créez des services, ajoutez uniquement des fournisseurs de capacité à la stratégie de fournisseur de capacité capables d'exécuter la tâche sur le plus petit type d'instance du groupe Auto Scaling. Pour les autres paramètres, vous pouvez utiliser des contraintes de placement.

- Pour chaque groupe de tâches, Amazon ECS calcule le nombre d'instances requises pour exécuter les tâches non placées. Ce calcul utilise une stratégie `binpack`. Cette stratégie tient compte du vCPU, de la mémoire, des interfaces réseau élastiques (ENI), des ports et

des GPUs exigences des tâches. Il tient également compte de la disponibilité des ressources des EC2 instances Amazon. Les valeurs des types d'instance les plus grands sont traitées comme le nombre maximal d'instances calculé. Les valeurs du plus petit type d'instance sont utilisées comme protection. Si le type d'instance le plus petit ne peut pas exécuter au moins une instance de la tâche, le calcul considère la tâche comme non compatible. Par conséquent, la tâche est exclue du calcul de montée en puissance. Lorsque toutes les tâches ne sont pas compatibles avec le plus petit type d'instance, l'autoscaling du cluster s'arrête et la valeur `CapacityProviderReservation` reste à `targetCapacity`.

- Amazon ECS publie la `CapacityProviderReservation` métrique CloudWatch par rapport à `minimumScalingStepSize` si l'une des conditions suivantes est le cas.
  - Le nombre maximal d'instances calculé est inférieur à la taille minimale de l'étape de mise à l'échelle.
  - La valeur inférieure du nombre d'instances calculé `maximumScalingStepSize` ou du nombre maximal d'instances calculé.
- CloudWatch les alarmes utilisent la `CapacityProviderReservation` métrique pour les fournisseurs de capacité. Lorsque la métrique `CapacityProviderReservation` est supérieure à la valeur `targetCapacity`, les alarmes augmentent également la `DesiredCapacity` du groupe Auto Scaling. La `targetCapacity` valeur est un paramètre du fournisseur de capacité envoyé à l' CloudWatch alarme pendant la phase d'activation du dimensionnement automatique du cluster.

  La valeur par défaut `targetCapacity` est 100 %.
- Le groupe Auto Scaling lance des EC2 instances supplémentaires. Pour éviter le surprovisionnement, Auto Scaling s'assure que la capacité des EC2 instances récemment lancées est stabilisée avant de lancer de nouvelles instances. La scalabilité automatique vérifie si toutes les instances existantes ont passé la `instanceWarmupPeriod` (maintenant moins l'heure de lancement de l'instance). Le scale-out est bloqué pour les instances situées dans le. `instanceWarmupPeriod`

  Le nombre de secondes par défaut pour qu'une instance nouvellement lancée se prépare est de 300.

Pour plus d'informations, consultez Exploration approfondie de la mise à l'échelle automatique du cluster Amazon ECS.

Considérations relatives aux montées en puissance

Tenez compte des points suivants pour le processus de montée en puissance :

- Bien qu'il existe plusieurs contraintes de placement, nous vous recommandons d'utiliser uniquement la contrainte de placement des tâches `distinctInstance`. Celle-ci empêche l'arrêt du processus de montée en puissance, car vous utilisez une contrainte de placement qui n'est pas compatible avec les instances échantillonnées.
- La mise à l'échelle gérée fonctionne mieux si votre groupe Auto Scaling utilise les mêmes types d'instance ou des types d'instance similaires.
- Lorsqu'un processus de mise à l'échelle horizontale est requis et qu'aucune instance de conteneur n'est en cours d'exécution, Amazon ECS monte toujours en puissance à deux instances dans un premier temps, puis exécute des processus de mise à l'échelle horizontale ou de montée en puissance supplémentaires. Toute montée en puissance attend la période de préparation de l'instance. Pour la mise à l'échelle horizontale, Amazon ECS attend 15 minutes après une montée en puissance avant de lancer des processus de mise à l'échelle horizontale à tout moment.
- La deuxième étape de montée en puissance doit attendre l'expiration du délai de la `instanceWarmupPeriod`, ce qui peut affecter la limite d'échelle globale. Si vous devez réduire ce délai, assurez-vous qu'il `instanceWarmupPeriod` est suffisamment long pour que l' EC2instance puisse lancer et démarrer l'agent Amazon ECS (afin d'éviter le surprovisionnement).
- L'autoscaling de cluster prend en charge la configuration du lancement, les modèles de lancement et plusieurs types d'instances dans le groupe Auto Scaling du fournisseur de capacité. Vous pouvez également utiliser la sélection de type d'instance basée sur des attributs sans plusieurs types d'instances.
- Lorsque vous utilisez un groupe Auto Scaling avec des instances à la demande et plusieurs types d'instance ou instances Spot, placez les types d'instances plus importants plus haut dans la liste des priorités et ne spécifiez pas de pondération. La spécification d'une pondération n'est pas prise en charge pour le moment. Pour plus d'informations, consultez Groupes Auto Scaling avec types d'instance multiples dans le Guide de l'utilisateur AWS Auto Scaling .
- Amazon ECS lance ensuite la `minimumScalingStepSize`, si le nombre maximal d'instances calculé est inférieur à la taille minimale de l'étape de mise à l'échelle, ou la plus faible des valeurs entre `maximumScalingStepSize` et la valeur maximale calculée du nombre d'instances.
- Si un service Amazon ECS `run-task` lance une tâche et que les instances de conteneur du fournisseur de capacité ne disposent pas de suffisamment de ressources pour démarrer la tâche, Amazon ECS limite le nombre de tâches ayant ce statut pour chaque cluster et empêche toute

tâche de dépasser cette limite. Pour de plus amples informations, veuillez consulter Service Quotas.

Comportement de mise à l'échelle horizontale gérée

Amazon ECS surveille les instances de conteneur pour chaque fournisseur de capacité au sein du cluster. Lorsqu'une instance de conteneur n'exécute aucune tâche, elle est considérée comme vide et Amazon ECS démarre le processus de mise à l'échelle horizontale.

CloudWatch les alarmes scale-in nécessitent 15 points de données (15 minutes) avant que le processus de scale-in ne démarre pour le groupe Auto Scaling. Une fois que le processus de mise à l'échelle horizontale démarre jusqu'à ce qu'Amazon ECS ait besoin de réduire le nombre d'instances de conteneur enregistrées, le groupe Auto Scaling définit la `DesireCapacity` pour qu'elle soit supérieure à une instance et inférieure à 50 % par minute.

Lorsqu'Amazon ECS demande une montée en puissance (lorsque la `CapacityProviderReservation` est supérieure à 100) alors qu'un processus de mise à l'échelle horizontale est en cours, le processus de mise à l'échelle horizontale est arrêté et démarre dès le début si nécessaire.

Le comportement de mise à l'échelle horizontale est décrit plus en détail ci-dessous :

1. Amazon ECS calcule le nombre d'instances de conteneur vides. Une instance de conteneur est considérée comme vide même lorsque les tâches de démon sont exécutées.
2. Amazon ECS définit la valeur `CapacityProviderReservation` sur un nombre compris entre 0 et 100 qui utilise la formule suivante pour représenter le ratio entre la taille que doit avoir le groupe Auto Scaling par rapport à sa taille réelle, exprimé en pourcentage. Amazon ECS publie ensuite la métrique sur CloudWatch. Pour plus d'informations sur le calcul de la métrique, veuillez consulter le billet de blog Deep Dive on Amazon ECS Cluster Auto Scaling.

   ```
   CapacityProviderReservation = (number of instances needed) / (number of running
    instances) x 100
   ```

3. La `CapacityProviderReservation` métrique génère une CloudWatch alarme. Cette alarme met à jour la valeur `DesiredCapacity` pour le groupe Auto Scaling. Ensuite, l'une des actions suivantes se produit :
   - Si vous n'utilisez pas de terminaison gérée par le fournisseur de capacité, le groupe Auto Scaling sélectionne les EC2 instances conformément à la politique de résiliation du groupe Auto Scaling et met fin aux instances jusqu'à ce que le nombre d' EC2 instances atteigne

leDesiredCapacity. L'enregistrement des instances de conteneur est ensuite annulé du cluster.

- Si toutes les instances de conteneur utilisent la protection contre la résiliation gérée, Amazon ECS supprime la protection contre la mise à l'échelle horizontale sur les instances de conteneur qui sont vides. Le groupe Auto Scaling sera alors en mesure de mettre fin aux EC2 instances. L'enregistrement des instances de conteneur est ensuite annulé du cluster.

Contrôlez les instances auxquelles Amazon ECS met fin

> **Important**
>
> Vous devez activer la protection contre la mise à l'échelle horizontale d'instance Auto Scaling sur le groupe Auto Scaling pour utiliser la fonctionnalité de protection contre la résiliation gérée de l'autoscaling de cluster.

La protection gérée contre les interruptions permet le dimensionnement automatique du cluster afin de contrôler les instances qui sont résiliées. Lorsque vous avez utilisé la protection de résiliation gérée, Amazon ECS met fin uniquement aux EC2 instances sur lesquelles aucune tâche Amazon ECS n'est en cours d'exécution. Les tâches exécutées par un service utilisant la stratégie de planification DAEMON sont ignorées et une instance peut être résiliée par le biais de l'autoscaling de cluster, même lorsque l'instance exécute ces tâches. Cela est dû au fait que toutes les instances du cluster exécutent ces tâches.

Amazon ECS active d'abord l'option de protection évolutive des instances pour les EC2 instances du groupe Auto Scaling. Amazon ECS place ensuite les tâches sur les instances. Lorsque toutes les tâches autres que les démons sont arrêtées sur une instance, Amazon ECS lance le processus de scale-in et désactive la protection scale-in pour l'instance. EC2 Le groupe Auto Scaling peut ensuite résilier l'instance.

La protection évolutive des instances Auto Scaling contrôle les EC2 instances qui peuvent être résiliées par Auto Scaling. Les instances dont la fonction de mise à l'échelle horizontale est activée ne peuvent pas être résiliées pendant le processus de mise à l'échelle horizontale. Pour plus d'informations sur la protection évolutive des instances Auto Scaling, consultez la section Utilisation de la protection évolutive des instances dans le manuel Amazon EC2 Auto Scaling User Guide.

Vous pouvez définir le targetCapacity pourcentage de manière à disposer de capacités inutilisées. Cela permet de lancer les tâches futures plus rapidement, car le groupe Auto Scaling

n'a pas à lancer d'autres instances. Amazon ECS utilise la valeur de capacité cible pour gérer la CloudWatch métrique créée par le service. Amazon ECS gère la CloudWatch métrique. Le groupe Auto Scaling est traité comme un état stable, de sorte qu'aucune action de dimensionnement n'est requise. Les valeurs peuvent être comprises entre 0 et 100 %. Par exemple, pour configurer Amazon ECS de manière à conserver 10 % de capacité gratuite en plus de celle utilisée par les tâches Amazon ECS, définissez la valeur de capacité cible à 90 %. Tenez compte des points suivants lors de la définition de la valeur `targetCapacity` sur un fournisseur de capacité.

- Une `targetCapacity` valeur inférieure à 100 % représente la quantité de capacité libre (EC2 instances Amazon) qui doit être présente dans le cluster. La capacité libre signifie qu'il n'y a aucune tâche en cours d'exécution.
- Les contraintes de placement telles que les zones de disponibilité, sans `binpack` supplémentaire, forcent Amazon ECS à exécuter une tâche pour chaque instance, ce qui peut ne pas être le comportement souhaité.

Vous devez activer la protection contre la mise à l'échelle horizontale d'instance Auto Scaling sur le groupe Auto Scaling pour utiliser la protection contre la résiliation gérée. Si vous n'activez pas la protection contre la mise à l'échelle horizontale, l'activation de la protection contre la résiliation gérée peut entraîner un comportement indésirable. Par exemple, certaines instances peuvent être bloquées à l'état de drainage. Pour plus d'informations, consultez Using instance Scale-in protection dans le manuel Amazon EC2 Auto Scaling User Guide.

Lorsque vous utilisez la protection contre la résiliation auprès d'un fournisseur de capacité, n'effectuez aucune action manuelle, telle que le détachement de l'instance, sur le groupe Auto Scaling associé au fournisseur de capacité. Les actions manuelles peuvent interrompre la mise à l'échelle horizontale du fournisseur de capacité. Si vous détachez une instance du groupe Auto Scaling, vous devez également annuler l'enregistrement de l'instance détachée du cluster Amazon ECS.

Mise à jour de la protection de résiliation gérée pour les fournisseurs de capacité Amazon ECS

Lorsque vous utilisez la protection gérée contre les résiliations, vous devez mettre à jour les paramètres des fournisseurs de capacité existants.

console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.

3. Sur la page du cluster, sélectionnez l'onglet Infrastructure.
4. Choisissez le fournisseur de capacité.
5. Choisissez Mettre à jour pour modifier les paramètres du fournisseur de capacité.
6. Dans les paramètres du groupe Auto Scaling, activez ou désactivez la fonction Managed Terminaison Protection pour activer ou désactiver la fonctionnalité.
7. Choisissez Mettre à jour.

AWS CLI

Vous pouvez mettre à jour le paramètre de protection gérée contre les résiliations d'un fournisseur de capacité à l'aide de la `update-capacity-provider` commande suivante :

Pour activer la protection gérée contre les résiliations :

```
aws ecs update-capacity-provider \
  --capacity-provider CapacityProviderName \
  --auto-scaling-group-provider
 "managedScaling={status=ENABLED,targetCapacity=70,minimumScalingStepSize=1,maximumScalingStepS
```

Pour désactiver la protection gérée contre les résiliations :

```
aws ecs update-capacity-provider \
  --capacity-provider CapacityProviderName \
  --auto-scaling-group-provider
 "managedScaling={status=ENABLED,targetCapacity=70,minimumScalingStepSize=1,maximumScalingStepS
```

> **Note**
>
> Quelques minutes peuvent être nécessaires pour que les modifications prennent effet dans l'ensemble de votre cluster. Lorsque vous activez la protection gérée contre les interruptions de service, les instances qui exécutent déjà des tâches seront protégées contre les événements d'extension. Lors de la désactivation de la protection gérée contre les interruptions, l'indicateur de protection sera supprimé des instances lors du prochain cycle de gestion du fournisseur de capacité ECS.

Console pour exécuter des tâches

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page du cluster, sélectionnez l'onglet Tâches.
4. Choisissez la tâche.
5. Sous Configuration, activez ou désactivez la protection de résiliation gérée pour activer ou désactiver la fonctionnalité.
6. Choisissez Configurer la protection intégrée des tâches.

   La boîte de dialogue Configurer la protection intégrée des tâches s'affiche

   a. Sous Protection intégrée des tâches, activez l'option Activer.
   b. Pour Expire en minutes, entrez le nombre de minutes avant la fin de la protection intégrée des tâches.
   c. Choisissez Update (Mettre à jour)

Activation de la mise à l'échelle automatique du cluster Amazon ECS

Vous activez le dimensionnement automatique du cluster afin qu'Amazon ECS gère le dimensionnement des EC2 instances Amazon enregistrées dans votre cluster.

Si vous souhaitez utiliser la console pour activer le dimensionnement automatique du cluster, reportez-vous à la sectionCréation d'un fournisseur de capacité pour Amazon ECS.

Avant de commencer, créez un groupe Auto Scaling et un fournisseur de capacité. Pour de plus amples informations, veuillez consulter the section called “Fournisseurs de capacité pour le type de EC2 lancement”.

Pour activer le dimensionnement automatique du cluster, vous associez le fournisseur de capacité au cluster, puis vous activez le dimensionnement automatique du cluster.

1. Utilisez la commande `put-cluster-capacity-providers` pour associer un ou plusieurs fournisseurs de capacité au cluster.

   Pour ajouter les fournisseurs de AWS Fargate capacité, incluez les fournisseurs de `FARGATE_SPOT` capacité `FARGATE` et les fournisseurs de capacité dans la demande. Pour

plus d'informations, consultez `put-cluster-capacity-providers` dans la Référence des commandes de l'AWS CLI .

```
aws ecs put-cluster-capacity-providers \
  --cluster ClusterName \
  --capacity-providers CapacityProviderName FARGATE FARGATE_SPOT \
  --default-capacity-provider-strategy capacityProvider=CapacityProvider,weight=1
```

Pour ajouter un groupe Auto Scaling pour le type de EC2 lancement, incluez le nom du groupe Auto Scaling dans la demande. Pour plus d'informations, consultez `put-cluster-capacity-providers` dans la Référence des commandes de l'AWS CLI .

```
aws ecs put-cluster-capacity-providers \
  --cluster ClusterName \
  --capacity-providers CapacityProviderName \
  --default-capacity-provider-strategy capacityProvider=CapacityProvider,weight=1
```

2. Utilisez la commande `describe-clusters` pour vérifier que l'association a réussi. Pour plus d'informations, consultez `describe-clusters` dans la Référence des commandes de l'AWS CLI .

```
aws ecs describe-clusters \
  --cluster ClusterName \
  --include ATTACHMENTS
```

3. Utilisez la commande `update-capacity-provider` pour activer la mise à l'échelle automatique gérée pour le fournisseur de capacité. Pour plus d'informations, consultez `update-capacity-provider` dans la Référence des commandes de l'AWS CLI .

```
aws ecs update-capacity-provider \
  --capacity-providers CapacityProviderName \
  --auto-scaling-group-provider managedScaling=ENABLED
```

Désactiver le dimensionnement automatique du cluster Amazon ECS

Vous désactivez le dimensionnement automatique du cluster lorsque vous avez besoin d'un contrôle plus précis des EC2 instances enregistrées dans votre cluster,

Pour désactiver le dimensionnement automatique d'un cluster, vous pouvez soit dissocier le fournisseur de capacité dont le dimensionnement géré est activé du cluster, soit mettre à jour le fournisseur de capacité pour désactiver le dimensionnement géré.

Dissocier le fournisseur de capacité

Effectuez les étapes suivantes pour dissocier le fournisseur de capacité d'un cluster.

1. Utilisez la commande `put-cluster-capacity-providers` pour dissocier le fournisseur de capacité de groupe Auto Scaling du cluster. Le cluster peut conserver l'association avec les fournisseurs AWS Fargate de capacité. Pour plus d'informations, consultez `put-cluster-capacity-providers` dans la Référence des commandes de l'AWS CLI .

```
aws ecs put-cluster-capacity-providers \
  --cluster ClusterName \
  --capacity-providers FARGATE FARGATE_SPOT \
  --default-capacity-provider-strategy '[]'
```

   Utilisez la commande `put-cluster-capacity-providers` pour dissocier le fournisseur de capacité de groupe Auto Scaling du cluster. Pour plus d'informations, consultez `put-cluster-capacity-providers` dans la Référence des commandes de l'AWS CLI .

```
aws ecs put-cluster-capacity-providers \
  --cluster ClusterName \
  --capacity-providers [] \
  --default-capacity-provider-strategy '[]'
```

2. Utilisez la commande `describe-clusters` pour vérifier que la dissociation a réussi. Pour plus d'informations, consultez la section `describe-clusters` dans la référence des commandes AWS CLI .

```
aws ecs describe-clusters \
  --cluster ClusterName \
  --include ATTACHMENTS
```

Désactivez la mise à l'échelle gérée pour le fournisseur de capacité

Effectuez les étapes suivantes pour désactiver la mise à l'échelle gérée pour le fournisseur de capacité.

- Utilisez la commande `update-capacity-provider` pour désactiver la mise à l'échelle automatique gérée pour le fournisseur de capacité. Pour plus d'informations, consultez `update-capacity-provider` dans la Référence des commandes de l'AWS CLI .

```
aws ecs update-capacity-provider \
  --capacity-providers CapacityProviderName \
  --auto-scaling-group-provider managedScaling=DISABLED
```

Création d'un fournisseur de capacité pour Amazon ECS

Une fois la création du cluster terminée, vous pouvez créer un nouveau fournisseur de capacité (groupe Auto Scaling) pour le type de EC2 lancement. Les fournisseurs de capacité vous aident à gérer et à faire évoluer votre infrastructure pour vos applications.

Avant de créer le fournisseur de capacité, vous devez créer un groupe Auto Scaling. Pour plus d'informations, consultez la section Groupes Auto Scaling dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling.

Pour créer un fournisseur de capacité pour le cluster (console Amazon ECS)

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur la *name* page Cluster :, choisissez Infrastructure, puis Create.
5. Sur la page Create capacity provider (Créer des fournisseurs de capacité), configurez les options suivantes.

   a. Sous Basic details (Détails de base), pour le Capacity provider name (Nom du fournisseur de capacité), entrez un nom de fournisseur de capacité.

   b. Sous Auto Scaling group (Groupe Auto Scaling), pour Use an existing Auto Scaling group (Utiliser un groupe Auto Scaling existant), choisissez le groupe Auto Scaling.

   c. (Facultatif) Pour configurer une stratégie de dimensionnement, sous Stratégies de dimensionnement, configurez les options suivantes.

      - Pour qu'Amazon ECS gère les actions de mise à l'échelle horizontale et de montée en puissance, sélectionnez Turn on managed scaling (Activer la mise à l'échelle gérée).

- Pour empêcher la fermeture d'une EC2 instance avec des tâches Amazon ECS en cours d'exécution, sélectionnez Activer la protection contre le dimensionnement.
- Pour Définir la capacité cible, entrez la valeur cible pour la CloudWatch métrique utilisée dans la politique de dimensionnement du suivi des cibles gérée par Amazon ECS.

6. Choisissez Créer.

Mettre à jour un fournisseur de capacité Amazon ECS

Lorsque vous utilisez un groupe Auto Scaling comme fournisseur de capacité, vous pouvez modifier la stratégie de dimensionnement du groupe.

Pour mettre à jour un fournisseur de capacité pour le cluster (console Amazon ECS)

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur la *name* page Cluster :, choisissez Infrastructure, puis Update.
5. Sur la page Create capacity provider (Créer des fournisseurs de capacité), configurez les options suivantes.

   - Sous Groupe Auto Scaling de Scaling policies (Stratégies de dimensionnement), configurez les options suivantes.

     - Pour qu'Amazon ECS gère les actions de mise à l'échelle horizontale et de montée en puissance, sélectionnez Turn on managed scaling (Activer la mise à l'échelle gérée).
     - Pour éviter que les EC2 instances comportant des tâches Amazon ECS en cours d'exécution ne soient mises hors service, sélectionnez Activer la protection contre le dimensionnement.
     - Pour Définir la capacité cible, entrez la valeur cible pour la CloudWatch métrique utilisée dans la politique de dimensionnement du suivi des cibles gérée par Amazon ECS.

6. Choisissez Mettre à jour.

Supprimer un fournisseur de capacité Amazon ECS

Si vous avez fini d'utiliser un fournisseur de capacité de groupe Auto Scaling, vous pouvez le supprimer. Une fois le groupe supprimé, le fournisseur de capacité du groupe Auto Scaling passe à

l'`INACTIVE`état. Les fournisseurs de capacité ayant un état `INACTIVE` peuvent rester détectables dans votre compte pendant un certain temps. Cependant, cela est susceptible de changer à terme. Il est donc important de ne pas compter sur la persistance des fournisseurs de capacité dont l'état est `INACTIVE`. Avant que le fournisseur de capacité de groupe Auto Scaling ne soit supprimé, il doit être retiré de la stratégie de fournisseur de capacité de tous les services. Vous pouvez utiliser l'API `UpdateService` ou le flux de service de mise à jour dans la console Amazon ECS pour supprimer un fournisseur de capacité de la stratégie de fournisseur de capacité d'un service. Utilisez la nouvelle option de déploiement Forcer pour vous assurer que toutes les tâches utilisant la capacité d' EC2 instance Amazon fournie par le fournisseur de capacité sont transférées de manière à utiliser la capacité des fournisseurs de capacité restants.

Pour supprimer un fournisseur de capacité pour le cluster (console Amazon ECS)

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur la *`name`* page Cluster :, choisissez Infrastructure, le groupe Auto Scaling, puis choisissez Delete.
5. Dans le champ de confirmation, saisissez Supprimer ***`Auto Scaling group name`***
6. Sélectionnez Delete (Supprimer).

Arrêtez en toute sécurité les charges de travail Amazon ECS exécutées sur les instances EC2

Le drainage géré des instances facilite la résiliation progressive des instances Amazon EC2 . Cela permet à vos charges de travail de s'arrêter en toute sécurité et d'être reprogrammées vers des instances non résilientes. La maintenance et les mises à jour de l'infrastructure sont effectuées sans craindre de perturber les charges de travail. En utilisant le drainage d'instance géré, vous simplifiez les flux de travail de gestion de votre infrastructure qui nécessitent le remplacement des EC2 instances Amazon, tout en garantissant la résilience et la disponibilité de vos applications.

Le drainage d'instance géré par Amazon ECS fonctionne avec les remplacements d'instances de groupe Auto Scaling. Sur la base de l'actualisation des instances et de leur durée de vie maximale, les clients peuvent s'assurer qu'ils restent conformes aux dernières réglementations en matière de système d'exploitation et de sécurité en matière de capacité.

Le drainage d'instance géré ne peut être utilisé qu'avec les fournisseurs de capacité Amazon ECS. Vous pouvez activer le drainage géré des instances lorsque vous créez ou mettez à jour les fournisseurs de capacité de votre groupe Auto Scaling à l'aide de la console Amazon ECS ou du SDK. AWS CLI

Les événements suivants sont couverts par le drainage d'instance géré par Amazon ECS.

- Actualisation de l'instance du groupe Auto Scaling - Utilisez l'actualisation des instances pour effectuer le remplacement progressif de vos EC2 instances Amazon dans votre groupe Auto Scaling au lieu de le faire manuellement par lots. Cela est utile lorsque vous devez remplacer un grand nombre d'instances. L'actualisation d'une instance est initiée via la EC2 console Amazon ou l'`StartInstanceRefresh`API. Assurez-vous de sélectionner `Replace` la protection Scale-in lorsque vous appelez `StartInstanceRefresh` si vous utilisez une protection de résiliation gérée.
- Durée de vie maximale des instances - Vous pouvez définir une durée de vie maximale lorsqu'il s'agit de remplacer les instances du groupe Auto Scaling. Cela est utile pour planifier des instances de remplacement en fonction des politiques de sécurité internes ou de la conformité.
- Mise à l'échelle du groupe Auto Scaling - Sur la base des politiques de dimensionnement et des actions de dimensionnement planifiées, le groupe Auto Scaling prend en charge le dimensionnement automatique des instances. En utilisant un groupe Auto Scaling comme fournisseur de capacité Amazon ECS, vous pouvez intégrer des instances de groupe Auto Scaling lorsqu'aucune tâche n'y est exécutée.
- Contrôles de santé du groupe Auto Scaling - Le groupe Auto Scaling prend en charge de nombreux bilans de santé pour gérer la résiliation d'instances défectueuses.
- AWS CloudFormation mises à jour de la pile - Vous pouvez ajouter un `UpdatePolicy` attribut à votre AWS CloudFormation pile pour effectuer des mises à jour continues lorsque le groupe change.
- Rééquilibrage de la capacité ponctuelle - Le groupe Auto Scaling essaie de remplacer de manière proactive les instances ponctuelles présentant un risque d'interruption plus élevé sur la base de l'avis de rééquilibrage des EC2 capacités d'Amazon. Le groupe Auto Scaling met fin à l'ancienne instance lorsque la solution de remplacement est lancée et saine. Le drainage d'instance géré par Amazon ECS draine l'instance Spot de la même manière qu'une instance non Spot.
- Interruption ponctuelle - Les instances ponctuelles sont résiliées avec un préavis de deux minutes. Le drainage d'instance géré par Amazon ECS met l'instance en état de vidange en réponse.

Le cycle de vie d'Amazon EC2 Auto Scaling s'accroche à la gestion du drainage des instances

Les hooks du cycle de vie du groupe Auto Scaling permettent au client de créer des solutions déclenchées par certains événements du cycle de vie de l'instance et d'effectuer une action personnalisée lorsque cet événement se produit. Un groupe Auto Scaling autorise jusqu'à 50 hooks. Plusieurs hooks de terminaison peuvent exister et sont exécutés en parallèle, et le groupe Auto Scaling attend que tous les hooks soient terminés avant de terminer une instance.

Outre la terminaison de crochet gérée par Amazon ECS, vous pouvez également configurer vos propres crochets de terminaison de cycle de vie. Les hooks du cycle de vie ont un`default action`, et nous vous recommandons de le définir `continue` par défaut pour garantir que les autres hooks, tels que le hook géré par Amazon ECS, ne soient pas affectés par les erreurs provenant des hooks personnalisés.

Si vous avez déjà configuré un hook de fin de cycle de vie de groupe Auto Scaling et que vous avez également activé le drainage d'instance géré par Amazon ECS, les deux hooks de cycle de vie sont exécutés. Les horaires relatifs ne sont toutefois pas garantis. Les hooks du cycle de vie disposent d'un `default action` paramètre qui spécifie l'action à effectuer lorsque le délai d'expiration est expiré. En cas d'échec, nous vous recommandons d'utiliser `continue` comme résultat par défaut votre hook personnalisé. Cela garantit que les autres hooks, en particulier les hooks gérés par Amazon ECS, ne sont pas affectés par des erreurs dans votre hook de cycle de vie personnalisé. Le résultat alternatif de `abandon` fait sauter tous les autres crochets et doit être évité. Pour plus d'informations sur les hooks du cycle de vie des groupes Auto Scaling, consultez les hooks du cycle de vie Amazon EC2 Auto Scaling dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling.

Tâches et vidange d'instance géré

Le drainage d'instance géré par Amazon ECS utilise la fonctionnalité de drainage existante présente dans les instances de conteneur. La fonction de vidange des instances de conteneur remplace et arrête les tâches de réplication appartenant à un service Amazon ECS. Une tâche autonome, telle qu'une tâche invoquée par`RunTask`, qui est à l'`RUNNING`état `PENDING` ou reste inchangée. Vous devez attendre qu'ils soient terminés ou les arrêter manuellement. L'instance de conteneur reste dans `DRAINING` cet état jusqu'à ce que toutes les tâches soient arrêtées ou que 48 heures se soient écoulées. Les tâches du démon sont les dernières à s'arrêter une fois toutes les tâches de réplication arrêtées.

Drainage des instances géré et protection gérée contre les résiliations

Le drainage d'instance géré fonctionne même si la terminaison gérée est désactivée. Pour plus d'informations sur la gestion de la protection contre les licenciements, consultezContrôlez les instances auxquelles Amazon ECS met fin.

Le tableau suivant résume le comportement des différentes combinaisons de terminaison gérée et de drainage géré.

Résiliation gérée	Drainage géré	Outcome
Activées	Activées	Amazon ECS protège les EC2 instances Amazon qui exécutent des tâches contre les interruptions dues à des événements de scale-in. Toutes les instances en cours de résiliation, telles que celles pour lesquelles la protection contre la résiliation n'est pas définie, celles qui ont subi une

Résiliation gérée	Drainage géré	Outcome
		interruption ponctuelle ou qui sont forcées par une actualisation de l'instance, sont correctement drainées.
Désactivées	Activées	Amazon ECS ne protège pas les EC2 instances Amazon exécutant des tâches contre l'évolutivité. Cependant, toutes les instances mises hors service sont correctement vidées.

Résiliation gérée	Drainage géré	Outcome
Activé	Désactivées	Amazon ECS protège les EC2 instances Amazon qui exécutent des tâches contre les interruptions dues à des événements de scale-in. Cependant, les instances peuvent toujours être résiliées en cas d'interruption ponctuelle ou d'actualisation forcée de l'instance, ou si elles n'exécutent aucune

Résiliation gérée	Drainage géré	Outcome
		tâche. Amazon ECS n'effectue pas de vidange progressive pour ces instances et lance des tâches de service de remplacement après leur arrêt.

Résiliation gérée	Drainage géré	Outcome
Désactivées	Désactivées	EC2 Les instances Amazon peuvent être étendues ou résiliées à tout moment, même si elles exécutent des tâches Amazon ECS. Amazon ECS lancera les tâches de service de remplacement après leur arrêt.

Drainage d'instance géré et vidange d'instance Spot

Avec le drainage d'une instance Spot, vous pouvez définir une variable d'environnement `ECS_ENABLE_SPOT_INSTANCE_DRAINING` sur l'agent Amazon ECS qui permet à Amazon ECS de placer une instance en état de vidange en réponse à une interruption ponctuelle de deux minutes. Le drainage des instances géré par Amazon ECS facilite l'arrêt progressif des EC2 instances Amazon en cours de résiliation pour de nombreuses raisons, et pas simplement pour une interruption ponctuelle. Par exemple, vous pouvez utiliser le rééquilibrage de capacité d'Amazon EC2 Auto Scaling pour remplacer de manière proactive une instance Spot en cas de risque élevé d'interruption, et le drainage géré des instances permet d'arrêter progressivement l'instance Spot remplacée.

Lorsque vous utilisez le drainage d'instance géré, il n'est pas nécessaire d'activer le drainage d'instance Spot séparément. Par conséquent, `ECS_ENABLE_SPOT_INSTANCE_DRAINING` dans le groupe Auto Scaling, les données utilisateur sont redondantes. Pour plus d'informations sur le drainage des instances Spot, consultezInstances Spot.

Comment fonctionne le drainage géré des instances avec EventBridge

Les événements de vidange des instances gérées par Amazon ECS sont publiés sur Amazon EventBridge, et Amazon ECS crée une règle EventBridge gérée dans le bus par défaut de votre compte pour prendre en charge le drainage des instances gérées. Vous pouvez filtrer ces événements vers d'autres AWS services tels que Lambda, Amazon SNS et Amazon SQS à des fins de surveillance et de résolution des problèmes.

- Amazon EC2 Auto Scaling envoie un événement EventBridge lorsqu'un hook du cycle de vie est invoqué.
- Les avis d'interruption ponctuelle sont publiés à EventBridge.
- Amazon ECS génère des messages d'erreur que vous pouvez récupérer via la console Amazon ECS et APIs.
- EventBridge comporte des mécanismes de réessai intégrés pour atténuer les défaillances temporaires.

Configuration des fournisseurs de capacité Amazon ECS pour arrêter les instances en toute sécurité

Vous pouvez activer le drainage géré des instances lorsque vous créez ou mettez à jour les fournisseurs de capacité de votre groupe Auto Scaling à l'aide de la console Amazon ECS et AWS CLI.

> **Note**
>
> Le drainage géré des instances est activé par défaut lorsque vous créez un fournisseur de capacité.

Voici des exemples d'utilisation du AWS CLI pour créer un fournisseur de capacité avec le drainage d'instance géré activé et pour activer le drainage d'instance géré pour le fournisseur de capacité existant d'un cluster.

Créez un fournisseur de capacité avec le drainage géré des instances activé

Pour créer un fournisseur de capacité avec le drainage d'instance géré activé, utilisez la `create-capacity-provider` commande. Définissez le paramètre `managedDraining` sur `ENABLED`.

```
aws ecs create-capacity-provider \
--name capacity-provider \
--auto-scaling-group-provider '{
  "autoScalingGroupArn": "asg-arn",
  "managedScaling": {
    "status": "ENABLED",
    "targetCapacity": 100,
    "minimumScalingStepSize": 1,
    "maximumScalingStepSize": 1
  },
  "managedDraining": "ENABLED",
  "managedTerminationProtection": "ENABLED",
}'
```

Réponse :

```
{
    "capacityProvider": {
        "capacityProviderArn": "capacity-provider-arn",
        "name": "capacity-provider",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "asg-arn",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
                "maximumScalingStepSize": 1
            },
            "managedTerminationProtection": "ENABLED"
            "managedDraining": "ENABLED"
        }
    }
}
```

Activer le drainage d'instance géré pour le fournisseur de capacité existant d'un cluster

Activez le drainage d'instance géré pour le fournisseur de capacité existant d'un cluster à l'aide de la `update-capacity-provider` commande. Vous voyez que cela dit `DISABLED` et `updateStatus` dit `managedDraining` actuellement`UPDATE_IN_PROGRESS`.

```
aws ecs update-capacity-provider \
--name cp-draining \
--auto-scaling-group-provider '{
  "managedDraining": "ENABLED"
}
```

Réponse :

```
{
    "capacityProvider": {
        "capacityProviderArn": "cp-draining-arn",
        "name": "cp-draining",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "asg-draining-arn",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
                "maximumScalingStepSize": 1,
                "instanceWarmupPeriod": 300
            },
            "managedTerminationProtection": "DISABLED",
            "managedDraining": "DISABLED" // before update
        },
        "updateStatus": "UPDATE_IN_PROGRESS", // in progress and need describe again to
 find out the result
        "tags": [
        ]
    }
}
```

Utilisez la `describe-clusters` commande et incluez`ATTACHMENTS`. `status`L'instance gérée qui draine la pièce jointe est`PRECREATED`, et l'ensemble l'`attachmentsStatus`est`UPDATING`.

```
aws ecs describe-clusters --clusters cluster-name --include ATTACHMENTS
```

Réponse :

```
{
    "clusters": [
        {
            ...

            "capacityProviders": [
                "cp-draining"
            ],
            "defaultCapacityProviderStrategy": [],
            "attachments": [
                # new precreated managed draining attachment
                {
                    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                    "type": "managed_draining",
                    "status": "PRECREATED",
                    "details": [
                        {
                            "name": "capacityProviderName",
                            "value": "cp-draining"
                        },
                        {
                            "name": "autoScalingLifecycleHookName",
                            "value": "ecs-managed-draining-termination-hook"
                        }
                    ]
                },

                ...

            ],
            "attachmentsStatus": "UPDATING"
        }
    ],
    "failures": []
}
```

Lorsque la mise à jour est terminée, `describe-capacity-providers` utilisez-le, et vous `managedDraining` le verrez maintenant`ENABLED`.

```
aws ecs describe-capacity-providers --capacity-providers cp-draining
```

Réponse :

```
{
    "capacityProviders": [
        {
            "capacityProviderArn": "cp-draining-arn",
            "name": "cp-draining",
            "status": "ACTIVE",
            "autoScalingGroupProvider": {
                "autoScalingGroupArn": "asg-draning-arn",
                "managedScaling": {
                    "status": "ENABLED",
                    "targetCapacity": 100,
                    "minimumScalingStepSize": 1,
                    "maximumScalingStepSize": 1,
                    "instanceWarmupPeriod": 300
                },
                "managedTerminationProtection": "DISABLED",
                "managedDraining": "ENABLED" // successfully update
            },
            "updateStatus": "UPDATE_COMPLETE",
            "tags": []
        }
    ]
}
```

Résolution des problèmes liés au drainage des instances géré par Amazon ECS

Vous devrez peut-être résoudre les problèmes liés au drainage des instances gérées. Vous trouverez ci-dessous un exemple de problème et de résolution que vous pourriez rencontrer lors de son utilisation.

Les instances ne s'arrêtent pas après avoir dépassé leur durée de vie maximale lors de l'utilisation de la mise à l'échelle automatique.

Si vos instances ne s'arrêtent pas même après avoir atteint ou dépassé la durée de vie maximale lors de l'utilisation d'un groupe de dimensionnement automatique, c'est peut-être parce qu'elles sont protégées contre le scale-in. Vous pouvez désactiver la gestion de la résiliation et autoriser le drainage géré pour gérer le recyclage des instances.

Création de ressources pour le dimensionnement automatique du cluster Amazon ECS à l'aide du AWS Management Console

Découvrez comment créer les ressources pour le dimensionnement automatique des clusters à l'aide du AWS Management Console. Lorsque les ressources nécessitent un nom, nous utilisons le préfixe `ConsoleTutorial` pour garantir qu'elles ont toutes un nom unique et pour les rendre faciles à localiser.

Rubriques

- Prérequis
- Étape 1 : Créer un cluster Amazon ECS
- Étape 2 : Enregistrer une définition de tâche
- Étape 3 : Exécuter une tâche
- Étape 4 : Vérifier
- Étape 5 : nettoyer

Prérequis

Le didacticiel suppose de remplir les prérequis suivants :

- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique Amazon ECS_ FullAccess IAM.
- Le rôle IAM d'instance de conteneur Amazon ECS est créé. Pour de plus amples informations, veuillez consulter Rôle IAM d'instance de conteneur Amazon ECS.
- Le rôle IAM lié à un service Amazon ECS service est créé. Pour de plus amples informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS.
- Le rôle IAM lié à un service Auto Scaling est créé. Pour plus d'informations, consultez la section Service-Linked Roles for Amazon EC2 Auto Scaling dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling.
- Vous avez un créé un VPC et un groupe de sécurité prêts à être utilisés. Pour de plus amples informations, veuillez consulter the section called “Créer un Virtual Private Cloud”.

Étape 1 : Créer un cluster Amazon ECS

Pour créer un cluster Amazon ECS, effectuez les étapes suivantes.

Amazon ECS crée un modèle de lancement Amazon EC2 Auto Scaling et un groupe Auto Scaling en votre nom dans le cadre de la AWS CloudFormation pile.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, sélectionnez Clusters, puis Créer un cluster.
3. Sous Cluster configuration (Configuration de cluster), pour Cluster name (Nom du cluster), saisissez `ConsoleTutorial-cluster`.
4. Sous Infrastructure, désactivez AWS Fargate (sans serveur), puis sélectionnez Amazon instances. EC2 Ensuite, configurez le groupe Auto Scaling qui agit en tant que fournisseur de capacité.
   - Dans Groupe Auto Scaling (ASG). Sélectionnez Créer un ASG, puis fournissez les informations suivantes sur le groupe :
     - Pour Operating system/Architecture (Système d'exploitation/Architecture), choisissez Amazon Linux 2.
     - EC2 Par type d'instance, choisissez t3.nano.
     - Pour Capacity (Capacité), saisissez le nombre minimum et le nombre maximum d'instances à lancer dans le groupe Auto Scaling.
5. (Facultatif) Pour gérer les identifications de cluster, développez Tags (Identifications), puis effectuez l'une des opérations suivantes :

   [Add a tag] Choisissez Add tag (Ajouter une étiquette) et procédez comme suit :

   - Pour Key (Clé), saisissez le nom de la clé.
   - Pour Value (Valeur), saisissez la valeur de clé.

   [Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de l'étiquette.
6. Choisissez Créer.

Étape 2 : Enregistrer une définition de tâche

Avant de pouvoir exécuter une tâche sur votre cluster, vous devez enregistrer une définition de tâche. Les définitions de tâches sont des listes de conteneurs regroupés ensemble. L'exemple suivant est une définition de tâche simple qui utilise une image `amazonlinux` de Docker Hub et qui est simplement en veille. Pour plus d'informations sur les paramètres de définition des tâches disponibles, consultez Définitions de tâche Amazon ECS.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
4. Dans la zone de l'éditeur JSON, collez le contenu suivant.

```
{
    "family": "ConsoleTutorial-taskdef",
    "containerDefinitions": [
        {
            "name": "sleep",
            "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
            "memory": 20,
            "essential": true,
            "command": [
                "sh",
                "-c",
                "sleep infinity"
            ]
        }
    ],
    "requiresCompatibilities": [
        "EC2"
    ]
}
```

5. Choisissez Créer.

Étape 3 : Exécuter une tâche

Après avoir enregistré une définition de tâche pour votre compte, vous pouvez exécuter une tâche dans le cluster. Pour ce didacticiel, vous exécutez cinq instances de la définition de tâche `ConsoleTutorial-taskdef` de votre cluster `ConsoleTutorial-cluster`.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez ConsoleTutorial-cluster.
3. Dans Tâches, choisissez Exécuter une nouvelle tâche.
4. Dans la section Environnement, sous Options de calcul, choisissez Stratégie de fournisseur de capacité.
5. Sous Configuration de déploiement, pour Type d'application, sélectionnez Tâche.
6. Choisissez ConsoleTutorial-taskdef dans la liste déroulante Famille.
7. Sous Tâches souhaitées, saisissez 5.
8. Choisissez Créer.

Étape 4 : Vérifier

À ce stade du didacticiel, vous devez avoir un cluster avec cinq tâches en cours d'exécution et un groupe Auto Scaling avec un fournisseur de capacité. La mise à l'échelle gérée Amazon ECS est activée pour le fournisseur de capacité.

Nous pouvons vérifier que tout fonctionne correctement en consultant les CloudWatch métriques, les paramètres du groupe Auto Scaling et enfin le nombre de tâches du cluster Amazon ECS.

Pour consulter les CloudWatch statistiques de votre cluster

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans la barre de navigation, en haut de l'écran, sélectionnez la région .
3. Dans le panneau de navigation, sous Métriques, sélectionnez Toutes les métriques.
4. Sur la page Toutes les métriques, sous l'onglet Parcourir, sélectionnez `AWS/ECS/ManagedScaling`.
5. Choisissez CapacityProviderName, ClusterName.
6. Cochez la case correspondant au `ConsoleTutorial-cluster` ClusterName.
7. Dans l'onglet Graphique des métriques, faites passer la valeur de Période à 30 secondes et celle de Statistique à Maximum.

La valeur affichée dans le graphique indique la valeur de capacité cible pour le fournisseur de capacité. Elle doit commencer à `100`, ce qui correspond au pourcentage de capacité cible que nous avons défini. Elle doit ensuite monter à `200`, ce qui déclenche une alarme en vertu de la politique de suivi des objectifs et d'échelonnement. L'alarme déclenche alors la montée en puissance du groupe Auto Scaling.

Effectuez les étapes suivantes pour afficher les détails de votre groupe Auto Scaling et vérifier que la montée en puissance a bien eu lieu.

Pour vérifier que le groupe Auto Scaling est monté en puissance

1. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans la barre de navigation, en haut de l'écran, sélectionnez la région .
3. Dans le volet de navigation, sous Auto Scaling, choisissez Auto Scaling Groups (Groupes Auto Scaling).
4. Choisissez le groupe Auto Scaling `ConsoleTutorial-cluster` créé dans ce didacticiel. Consultez la valeur sous Capacité souhaitée et consultez les instances sous l'onglet Gestion des instances pour confirmer que votre groupe est monté en puissance de deux instances.

Suivez les étapes ci-dessous pour consulter votre cluster Amazon ECS afin de confirmer que les EC2 instances Amazon ont été enregistrées auprès du cluster et que vos tâches ont été transférées vers un `RUNNING` statut.

Vérifier les instances dans le groupe Auto Scaling

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster `ConsoleTutorial-cluster`.
4. Dans l'onglet Tâches, vérifiez qu'il y a bien cinq tâches à l'état `RUNNING`.

Étape 5 : nettoyer

Lorsque vous avez terminé ce didacticiel, nettoyez les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources que vous n'utilisez pas. La suppression des fournisseurs

de capacité et des définitions de tâches n'est pas prise en charge, mais ces ressources n'entraînent aucun coût.

Pour nettoyer les ressources du didacticiel

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez ConsoleTutorial-cluster.
4. Sur la page ConsoleTutorial-cluster, choisissez l'onglet Tâches, puis sélectionnez Arrêter, Tout arrêter.
5. Dans le panneau de navigation, choisissez Clusters.
6. Sur la page Clusters, choisissez ConsoleTutorial-cluster.
7. Dans le coin supérieur droit de la page, choisissez Supprimer le cluster.
8. Dans le champ de confirmation, entrez delete ConsoleTutorial-cluster et choisissez Supprimer.
9. Supprimez les groupes Auto Scaling en effectuant les étapes suivantes.
    a. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
    b. Dans la barre de navigation, en haut de l'écran, sélectionnez la région .
    c. Dans le volet de navigation, sous Auto Scaling, choisissez Auto Scaling Groups (Groupes Auto Scaling).
    d. Sélectionnez le groupe Auto Scaling `ConsoleTutorial-cluster`, puis sélectionnez Actions.
    e. Dans le menu Actions, choisissez Delete (Supprimer). Saisissez delete dans la zone de confirmation, puis choisissez Supprimer.

Instances de EC2 conteneur Amazon pour Amazon ECS

Une instance de conteneur Amazon ECS est une EC2 instance Amazon qui exécute l'agent de conteneur Amazon ECS et qui est enregistrée dans un cluster. Lorsque vous exécutez des tâches avec Amazon ECS en utilisant le type de EC2 lancement, le type de lancement externe ou un fournisseur de capacité de groupe Auto Scaling, vos tâches sont placées sur vos instances de conteneur actives. Vous êtes responsable de la gestion et de la maintenance des instances de conteneur.

Bien que vous puissiez créer votre propre AMI d' EC2 instance Amazon qui répond aux spécifications de base nécessaires pour exécuter vos charges de travail conteneurisées sur Amazon ECS, les AMI optimisées pour Amazon ECS AMIs sont préconfigurées et testées sur Amazon ECS par des ingénieurs. AWS C'est la façon la plus simple de démarrer et d'exécuter rapidement vos conteneurs sur AWS .

Lorsque vous créez un cluster à l'aide de la console, Amazon ECS crée un modèle de lancement pour vos instances avec la dernière AMI associée au système d'exploitation sélectionné.

Lorsque vous créez AWS CloudFormation un cluster, le paramètre SSM fait partie du modèle de EC2 lancement Amazon pour les instances du groupe Auto Scaling. Vous pouvez configurer le modèle pour utiliser un paramètre dynamique de Systems Manager afin de déterminer l'AMI optimisée Amazon ECS à déployer. Ce paramètre garantit que chaque fois que vous déployez la pile, elle vérifie si une mise à jour disponible doit être appliquée aux EC2 instances. Pour un exemple d'utilisation du paramètre Systems Manager, consultez la section Créer un cluster Amazon ECS avec l'AMI Amazon Linux 2023 optimisée pour Amazon ECS dans le guide de l'AWS CloudFormation utilisateur.

- Récupération des métadonnées de l'AMI Linux optimisées pour Amazon ECS
- Récupération des métadonnées d'AMI optimisées pour Amazon ECS Bottlerocket
- Récupération des métadonnées de l'AMI Windows optimisées pour Amazon ECS

Vous pouvez choisir parmi les types d'instances compatibles avec votre application. Avec des instances plus grandes, vous pouvez lancer plus de tâches en même temps. Avec des instances plus petites, vous pouvez effectuer une mise à l'échelle plus fine afin de réduire les coûts. Vous n'avez pas besoin de choisir un seul type d' EC2 instance Amazon adapté à toutes les applications de votre cluster. Au lieu de cela, vous pouvez créer plusieurs groupes Auto Scaling dans lesquels chaque groupe possède un type d'instance différent. Vous pouvez ensuite créer un fournisseur EC2 de capacité Amazon pour chacun de ces groupes.

Suivez les directives suivantes pour déterminer les types de famille d'instances et le type d'instance à utiliser :

- Éliminez les types ou familles d'instances qui ne répondent pas aux exigences spécifiques de votre application. Par exemple, si votre application nécessite un GPU, vous pouvez exclure tous les types d'instances qui n'en ont pas.
- Tenez compte des exigences, notamment en termes de débit réseau et de stockage.

- Tenez compte du processeur et de la mémoire. En règle générale, le processeur et la mémoire doivent être suffisamment volumineux pour contenir au moins une réplique de la tâche que vous souhaitez exécuter.

Instances Spot

La capacité Spot peut permettre de réaliser d'importantes économies par rapport aux instances à la demande. La capacité Spot est une capacité excédentaire dont le prix est nettement inférieur à celui de la capacité réservée ou à la demande. Elle convient aux charges de travail de traitement par lots et de machine learning, ainsi qu'aux environnements de développement et intermédiaire. Plus généralement, elle convient à toutes les charges de travail qui tolèrent des temps d'arrêt temporaires.

Ayez conscience des conséquences suivantes, car la capacité Spot peut ne pas être disponible en permanence.

- Pendant les périodes de très forte demande, la capacité Spot peut être indisponible. Cela peut retarder le lancement des instances Amazon EC2 Spot. Dans ces cas, les services Amazon ECS réessayent de lancer des tâches, et les groupes Amazon EC2 Auto Scaling réessayent également de lancer des instances, jusqu'à ce que la capacité requise soit disponible. Amazon EC2 ne remplace pas la capacité Spot par une capacité à la demande.
- Lorsque la demande globale de capacité augmente, les instances Spot et les tâches peuvent être interrompues avec un avertissement de deux minutes seulement. Une fois l'avertissement envoyé, les tâches doivent commencer à s'arrêter de manière ordonnée si nécessaire avant que l'instance ne soit complètement résiliée. Cela permet de réduire les risques d'erreurs. Pour plus d'informations sur un arrêt progressif, veuillez consulter le billet de blog Graceful shutdowns with ECS.

Pour réduire les pénuries de capacité Spot, tenez compte des recommandations suivantes :

- Utilisez plusieurs régions et zones de disponibilité : la capacité Spot varie en fonction de la région et de la zone de disponibilité. Vous pouvez améliorer la disponibilité Spot en exécutant vos charges de travail dans plusieurs régions et zones de disponibilité. Si possible, spécifiez des sous-réseaux dans toutes les zones de disponibilité des régions dans lesquelles vous exécutez vos tâches et instances.
- Utiliser plusieurs types d' EC2 instances Amazon : lorsque vous utilisez des politiques d'instance mixtes avec Amazon EC2 Auto Scaling, plusieurs types d'instances sont lancés dans votre groupe Auto Scaling. Cela garantit qu'une demande de capacité Spot peut être satisfaite en cas de besoin.

Pour optimiser la fiabilité et réduire la complexité, utilisez des types d'instances dotés à peu près de la même quantité de processeur et de mémoire dans votre stratégie d'instances mixtes. Ces instances peuvent provenir d'une génération différente ou de variantes du même type d'instance de base. Veuillez noter qu'elles peuvent être dotées de fonctionnalités supplémentaires dont vous n'aurez peut-être pas besoin. Une telle liste pourrait inclure, par exemple m4.large, m5.large, m5a.large, m5d.large, m5n.large, m5dn.large et m5ad.large. Pour plus d'informations, consultez la section Groupes Auto Scaling avec plusieurs types d'instances et options d'achat dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling.

- Utilisez la stratégie d'allocation ponctuelle optimisée en termes de capacité : avec Amazon EC2 Spot, vous pouvez choisir entre des stratégies d'allocation optimisées en termes de capacité et de coûts. Si vous optez pour la stratégie d'optimisation des capacités lors du lancement d'une nouvelle instance, Amazon EC2 Spot sélectionne le type d'instance le plus disponible dans la zone de disponibilité sélectionnée. Cela permet de réduire le risque que l'instance soit résiliée peu après son lancement.

Pour plus d'informations sur la façon de configurer les avis de résiliation ponctuels sur vos instances de conteneur, voir :

- Configuration des instances de conteneur Linux Amazon ECS pour recevoir des notifications relatives aux instances Spot
- Configuration des instances de conteneur Windows Amazon ECS pour recevoir des notifications relatives aux instances Spot

Linux optimisé pour Amazon ECS AMIs

> **Important**
>
> L'AMI Amazon Linux 2 optimisée pour Amazon ECS arrive end-of-life le 30 juin 2026, reflétant la même date de fin de vie que le système d'exploitation Amazon Linux 2 en amont (pour plus d'informations, consultez Amazon Linux 2). FAQs Nous encourageons les clients à mettre à niveau leurs applications pour utiliser Amazon Linux 2023, qui inclut un support à long terme jusqu'en 2028. Pour plus d'informations sur la migration d'Amazon Linux 2 vers Amazon Linux 2023, consultez Migration de l'AMI optimisée pour Amazon Linux 2 vers l'AMI optimisée pour Amazon Linux 2023 pour Amazon ECS.

Amazon ECS fournit des solutions optimisées pour Amazon ECS AMIs qui sont préconfigurées avec les exigences et les recommandations nécessaires pour exécuter vos charges de travail de conteneurs. Nous vous recommandons d'utiliser l'AMI Amazon Linux 2023 optimisée pour Amazon ECS pour vos instances Amazon EC2 . Le lancement de vos instances de conteneur à partir de l'AMI optimisée pour Amazon ECS la plus récente garantit que vous recevez la version actuelle de l'agent de conteneur ainsi que ses mises à jour de sécurité. Pour plus d'informations sur le lancement d'une instance, veuillez consulter Lancement d'une instance de conteneur Amazon ECS Linux.

Lorsque vous créez un cluster à l'aide de la console, Amazon ECS crée un modèle de lancement pour vos instances avec la dernière AMI associée au système d'exploitation sélectionné.

Lorsque vous créez AWS CloudFormation un cluster, le paramètre SSM fait partie du modèle de EC2 lancement Amazon pour les instances du groupe Auto Scaling. Vous pouvez configurer le modèle pour utiliser un paramètre dynamique de Systems Manager afin de déterminer l'AMI optimisée Amazon ECS à déployer. Ce paramètre garantit que chaque fois que vous déployez la pile, elle vérifie si une mise à jour disponible doit être appliquée aux EC2 instances. Pour un exemple d'utilisation du paramètre Systems Manager, consultez la section Créer un cluster Amazon ECS avec l'AMI Amazon Linux 2023 optimisée pour Amazon ECS dans le guide de l'AWS CloudFormation utilisateur.

Si vous devez personnaliser l'AMI optimisée pour Amazon ECS, consultez Amazon ECS Optimized AMI Build Recipes sur. GitHub

Les variantes suivantes de l'AMI optimisée pour Amazon ECS sont disponibles pour vos EC2 instances Amazon avec le système d'exploitation Amazon Linux 2023.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2023	AMI Amazon Linux 2023 optimisée pour Amazon ECS	Amazon Linux 2023 est la nouvelle génération d'Amazon Linux de AWS. Dans la plupart des cas, c'est recommandé pour lancer vos EC2 instances Amazon pour vos	Par défaut, l'AMI Amazon Linux 2023 optimisée pour Amazon ECS est fournie avec un volume racine unique de 30 Gio. Vous pouvez modifier la taille du volume

Système d'exploitation	AMI	Description	Configuration du stockage
		charges de travail Amazon ECS. Pour plus d'informations, veuillez consulter Qu'est-ce qu'Amazon Linux 2023 dans le Guide de l'utilisateur Amazon Linux 2023 (langue française non garantie).	racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation et pour les métadonnées et images Docker. Le système de fichiers par défaut pour l'AMI Amazon Linux 2023 optimisée pour Amazon ECS est `xfs`, tandis que Docker utilise le pilote de stockage `overlay2`. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2023 (arm64)	AMI Amazon Linux 2023 (arm64) optimisée pour Amazon ECS	Basée sur Amazon Linux 2023, il est recommandé d'utiliser cette AMI lors du lancement de vos EC2 instances Amazon, qui sont alimentées par des processeurs AWS Graviton/Graviton 2/Graviton 3/Graviton 4 basés sur ARM, pour vos charges de travail Amazon ECS. Pour plus d'informations, consultez les spécifications relatives aux instances Amazon à usage EC2 général dans le guide des types d' EC2 instances Amazon.	Par défaut, l'AMI Amazon Linux 2023 optimisée pour Amazon ECS est fournie avec un volume racine unique de 30 Gio. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation et pour les métadonnées et images Docker. Le système de fichiers par défaut pour l'AMI Amazon Linux 2023 optimisée pour Amazon ECS est `xfs`, tandis que Docker utilise le pilote de stockage `overlay2`. Pour plus d'informations, consultez Use the OverlayFS storage driver dans

Système d'exploitation	AMI	Description	Configuration du stockage
			la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2023 (Neuron)	AMI Amazon Linux 2023 optimisée pour Amazon ECS	Basé sur Amazon Linux 2023, ceci AMIis pour les instances Amazon EC2 Inf1, Trn1 ou Inf2. Il est préconfiguré avec les pilotes AWS Inferentia et AWS Trainium ainsi que le moteur d'exécution AWS Neuron pour Docker, qui facilite l'exécution des charges de travail d'inférence liées au machine learning sur Amazon ECS. Pour de plus amples informations, veuillez consulter Définitions de tâches Amazon ECS pour les charges de travail d'apprentissage automatique AWS Neuron. L'AMI Amazon Linux 2023 (Neuron) optimisée pour Amazon ECS n'est pas fournie avec le préinstallé. AWS CLI	Par défaut, l'AMI Amazon Linux 2023 optimisée pour Amazon ECS est fournie avec un volume racine unique de 30 Gio. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation et pour les métadonnées et images Docker. Le système de fichiers par défaut pour l'AMI Amazon Linux 2023 optimisée pour Amazon ECS est `xfs`, tandis que Docker utilise le pilote de stockage `overlay2`. Pour plus d'informations, consultez Use the OverlayFS storage driver dans

Système d'exploitation	AMI	Description	Configuration du stockage
			la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Processeur graphique Amazon Linux 2023	AMI GPU Amazon Linux 2023 optimisée pour Amazon ECS	Basée sur Amazon Linux 2023, il est recommandé d'utiliser cette AMI lors du lancement de vos instances EC2 basées sur le GPU Amazon pour vos charges de travail Amazon ECS. Il est préconfiguré avec des pilotes de noyau NVIDIA et un environnement d'exécution GPU Docker, ce qui permet d'exécuter des charges de travail exploitables sur GPUs Amazon ECS. Pour de plus amples informations, veuillez consulter Définitions de tâches Amazon ECS pour les charges de travail du GPU.	Par défaut, l'AMI Amazon Linux 2023 optimisée pour Amazon ECS est fournie avec un volume racine unique de 30 Gio. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation et pour les métadonnées et images Docker. Le système de fichiers par défaut pour l'AMI Amazon Linux 2023 optimisée pour Amazon ECS est `xfs`, tandis que Docker utilise le pilote de stockage `overlay2`. Pour plus d'informations, consultez Use the OverlayFS storage driver dans

Système d'exploitation	AMI	Description	Configuration du stockage
			la documentation Docker.

Les variantes suivantes de l'AMI optimisée pour Amazon ECS sont disponibles pour vos EC2 instances Amazon dotées du système d'exploitation Amazon Linux 2.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2	AMI Amazon Linux 2 noyau 5.10 optimisée pour Amazon ECS	Basée sur Amazon Linux 2, cette AMI est destinée à être utilisée lors du lancement de vos EC2 instances Amazon et vous souhaitez utiliser le noyau Linux 5.10 au lieu du noyau 4.14 pour vos charges de travail Amazon ECS. L'AMI Amazon Linux 2 noyau 5.10 optimisée pour Amazon ECS n'est pas fournie avec AWS CLI préinstallé.	Par défaut, les AMI Amazon ECS optimisées pour Amazon Linux 2 (AMI AMIs Amazon Linux 2 optimisée pour Amazon ECS, AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS et AMI optimisée pour le GPU Amazon ECS) sont livrées avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponibl e sur votre instance de conteneur. Ce stockage est utilisé

Système d'exploitation	AMI	Description	Configuration du stockage
			pour le système d'exploitation et pour les métadonnées et images Docker. Le système de fichiers par défaut pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS est `xfs`, tandis que Docker utilise le pilote de stockage `overlay2`. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2	AMI Amazon Linux 2 optimisée pour Amazon ECS	Destinée à vos charges de travail Amazon ECS. L'AMI Amazon Linux 2 optimisée pour Amazon ECS n'est pas fournie avec la AWS CLI préinstallée.	Par défaut, les AMI Amazon ECS optimisées pour Amazon Linux 2 (AMI AMIs Amazon Linux 2 optimisée pour Amazon ECS, AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS et AMI optimisée pour le GPU Amazon ECS) sont livrées avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponibl e sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation et pour les métadonnées et images Docker. Le système de fichiers par défaut pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS

Système d'exploitation	AMI	Description	Configuration du stockage
			est `xfs`, tandis que Docker utilise le pilote de stockage `overlay2`. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2 (arm64)	AMI Amazon Linux 2 noyau 5.10 (arm64) optimisée pour Amazon ECS	Basée sur Amazon Linux 2, cette AMI est destinée à vos EC2 instances Amazon, qui sont alimentées par AWS Graviton/Graviton 2/Graviton 3/Graviton 4 processeurs ARM, et vous souhaitez utiliser le noyau Linux 5.10 au lieu du noyau Linux 4.14 pour vos charges de travail Amazon ECS. Pour plus d'informations, consultez les spécifications relatives aux instances Amazon à usage EC2 général dans le guide des types d' EC2 instances Amazon. L'AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS n'est pas fournie avec le préinstallé. AWS CLI	Par défaut, les AMI Amazon ECS optimisées pour Amazon Linux 2 (AMI AMIs Amazon Linux 2 optimisée pour Amazon ECS, AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS et AMI optimisée pour le GPU Amazon ECS) sont livrées avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation et pour les métadonnées et images Docker. Le système de fichiers par défaut pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS

Système d'exploitation	AMI	Description	Configuration du stockage
			est `xfs`, tandis que Docker utilise le pilote de stockage `overlay2`. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2 (arm64)	AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS	Basée sur Amazon Linux 2, cette AMI est destinée à être utilisée lors du lancement de vos EC2 instances Amazon, qui sont alimentées par AWS Graviton/Graviton 2/Graviton 3/Graviton 4 processeurs ARM, pour vos charges de travail Amazon ECS. L'AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS n'est pas fournie avec le préinstallé. AWS CLI	Par défaut, les AMI Amazon ECS optimisées pour Amazon Linux 2 (AMI AMIs Amazon Linux 2 optimisée pour Amazon ECS, AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS et AMI optimisée pour le GPU Amazon ECS) sont livrées avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation et pour les métadonnées et images Docker. Le système de fichiers par défaut pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS

Système d'exploitation	AMI	Description	Configuration du stockage
			est `xfs`, tandis que Docker utilise le pilote de stockage `overlay2`. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2 (GPU)	AMI du noyau 5.10 optimisé pour le GPU Amazon ECS	Basée sur Amazon Linux 2, il est recommandé d'utiliser cette AMI lors du lancement de vos instances EC2 basées sur le GPU Amazon avec le noyau Linux 5.10 pour vos charges de travail Amazon ECS. Il est préconfiguré avec des pilotes de noyau NVIDIA et un environnement d'exécution GPU Docker, ce qui permet d'exécuter des charges de travail exploitables sur GPUs Amazon ECS. Pour de plus amples informations, veuillez consulter Définitions de tâches Amazon ECS pour les charges de travail du GPU.	Par défaut, les AMI Amazon ECS optimisées pour Amazon Linux 2 (AMI AMIs Amazon Linux 2 optimisée pour Amazon ECS, AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS et AMI optimisée pour le GPU Amazon ECS) sont livrées avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation et pour les métadonnées et images Docker. Le système de fichiers par défaut pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS

Système d'exploitation	AMI	Description	Configuration du stockage
			est `xfs`, tandis que Docker utilise le pilote de stockage `overlay2`. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2 (GPU)	AMI optimisée pour GPU Amazon ECS	Basée sur Amazon Linux 2, il est recommandé d'utiliser cette AMI lors du lancement de vos instances EC2 basées sur le GPU Amazon avec le noyau Linux 4.14 pour vos charges de travail Amazon ECS. Il est préconfiguré avec des pilotes de noyau NVIDIA et un environnement d'exécution GPU Docker, ce qui permet d'exécuter des charges de travail exploitables sur GPUs Amazon ECS. Pour de plus amples informations, veuillez consulter Définitions de tâches Amazon ECS pour les charges de travail du GPU.	Par défaut, les AMI Amazon ECS optimisées pour Amazon Linux 2 (AMI AMIs Amazon Linux 2 optimisée pour Amazon ECS, AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS et AMI optimisée pour le GPU Amazon ECS) sont livrées avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation et pour les métadonnées et images Docker. Le système de fichiers par défaut pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS

Système d'exploitation	AMI	Description	Configuration du stockage
			est `xfs`, tandis que Docker utilise le pilote de stockage `overlay2`. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2 (Neuron)	AMI Amazon Linux 2 (Neuron) 5.10 optimisé pour Amazon ECS	Basée sur Amazon Linux 2, cette AMI est destinée aux instances Amazon EC2 Inf1, Trn1 ou Inf2. Il est préconfiguré avec AWS Inferentia avec le noyau Linux 5.10 et les pilotes AWS Trainium, ainsi que le moteur d'exécution AWS Neuron pour Docker, qui facilite l'exécution des charges de travail d'inférence par apprentissage automatique sur Amazon ECS. Pour de plus amples informations, veuillez consulter Définitions de tâches Amazon ECS pour les charges de travail d'apprentissage automatique AWS Neuron. L'AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS n'est pas fournie avec	Par défaut, les AMI Amazon ECS optimisées pour Amazon Linux 2 (AMI AMIs Amazon Linux 2 optimisée pour Amazon ECS, AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS et AMI optimisée pour le GPU Amazon ECS) sont livrées avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation et pour les métadonnées et images Docker. Le système de fichiers par défaut pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS

Système d'exploitation	AMI	Description	Configuration du stockage
		le AWS CLI préinstallé.	est `xfs`, tandis que Docker utilise le pilote de stockage `overlay2`. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2 (Neuron)	AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS	Basée sur Amazon Linux 2, cette AMI est destinée aux instances Amazon EC2 Inf1, Trn1 ou Inf2. Il est préconfiguré avec les pilotes AWS Inferentia et AWS Trainium ainsi que le moteur d'exécution AWS Neuron pour Docker, qui facilite l'exécution des charges de travail d'inférence liées au machine learning sur Amazon ECS. Pour de plus amples informations, veuillez consulter Définitions de tâches Amazon ECS pour les charges de travail d'apprentissage automatique AWS Neuron. L'AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS n'est pas fournie avec le AWS CLI préinstallé.	Par défaut, les AMI Amazon ECS optimisées pour Amazon Linux 2 (AMI AMIs Amazon Linux 2 optimisée pour Amazon ECS, AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS et AMI optimisée pour le GPU Amazon ECS) sont livrées avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation et pour les métadonnées et images Docker. Le système de fichiers par défaut pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS

Système d'exploitation	AMI	Description	Configuration du stockage
			est `xfs`, tandis que Docker utilise le pilote de stockage `overlay2`. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Amazon ECS fournit un journal des modifications pour la variante Linux de l'AMI optimisée pour Amazon ECS sur. GitHub Pour plus d'informations, consultez Journal des modifications.

Les variantes Linux de l'AMI optimisée pour Amazon ECS utilisent l'AMI Amazon Linux 2 ou Amazon Linux 2023 comme base. Vous pouvez récupérer le nom de l'AMI pour chaque variante en interrogeant l'API du magasin de paramètres de Systems Manager. Pour de plus amples informations, veuillez consulter Récupération des métadonnées de l'AMI Linux optimisées pour Amazon ECS. Les notes de mise à jour de l'AMI Amazon Linux 2 sont également disponibles. Pour plus d'informations, consultez Notes de mise à jour Amazon Linux 2. Les notes de mise à jour d'Amazon Linux 2023 sont également disponibles. Pour plus d'informations, veuillez consulter Notes de mise à jour Amazon Linux 2023 (langue française non garantie).

Les pages suivantes fournissent des informations supplémentaires sur les modifications :

- Notes de mise à jour de l'AMI source sur GitHub
- Notes de mise à jour Docker Engine dans la documentation Docker
- Documentation du pilote NVIDIA dans la documentation NVIDIA
- Connectez-vous à la liste des modifications apportées à l'agent Amazon ECS GitHub

  Le code source de l'application `ecs-init`, ainsi que les scripts et la configuration nécessaires à l'empaquetage de l'agent, font désormais partie du référentiel de l'agent. Pour les anciennes versions `ecs-init` et emballages, consultez le journal des modifications Amazon ecs-init sur GitHub

Appliquer des mises à jour de sécurité à l'AMI optimisée pour Amazon ECS

Les versions optimisées pour Amazon ECS AMIs basées sur Amazon Linux contiennent une version personnalisée de. cloud-init Cloud-initest un package utilisé pour démarrer des images Linux dans un environnement de cloud computing et effectuer les actions souhaitées lors du lancement d'une instance. Par défaut, toutes les mises à jour de sécurité « critiques » et « importantes » sont appliquées dès le lancement de l'instance à toutes les mises à jour de sécurité optimisées pour Amazon ECS AMIs basées sur Amazon Linux publiées avant le 12 juin 2024.

À compter des versions du 12 juin 2024 de l'outil optimisé pour Amazon ECS AMIs basé sur Amazon Linux 2, le comportement par défaut n'inclura plus la mise à jour des packages au lancement. Nous vous recommandons plutôt de passer à une nouvelle AMI optimisée pour Amazon ECS au fur et à mesure que les versions seront disponibles. Les versions optimisées pour Amazon ECS AMIs sont publiées lorsque des mises à jour de sécurité ou des modifications de l'AMI de base sont disponibles. Cela garantira que vous recevez les dernières versions des packages et mises à jour de sécurité, et que les versions des packages sont immuables lors des lancements d'instances. Pour plus d'informations sur la récupération de la dernière AMI optimisée pour Amazon ECS, consultez. Récupération des métadonnées de l'AMI Linux optimisées pour Amazon ECS

Nous vous recommandons d'automatiser votre environnement pour qu'il soit mis à jour vers une nouvelle AMI dès qu'elle sera disponible. Pour plus d'informations sur les options disponibles, consultez Amazon ECS facilite la gestion des EC2 capacités, avec le drainage des instances géré.

Pour continuer à appliquer manuellement les mises à jour de sécurité « critiques » et « importantes » sur une version d'AMI, vous pouvez exécuter la commande suivante sur votre EC2 instance Amazon.

```
yum update --security
```

Si vous souhaitez réactiver les mises à jour de sécurité au lancement, vous pouvez ajouter la ligne suivante à la `#cloud-config` section des données utilisateur de cloud-init lors du lancement de votre instance Amazon. EC2 Pour plus d'informations, consultez la section Utilisation de cloud-init sur Amazon Linux 2 dans le guide de l'utilisateur Amazon Linux.

```
#cloud-config
repo_upgrade: security
```

Récupération des métadonnées de l'AMI Linux optimisées pour Amazon ECS

Vous pouvez récupérer par programmation les métadonnées de l'AMI optimisées pour Amazon ECS. Les métadonnées incluent le nom de l'AMI, la version de l'agent de conteneur Amazon ECS et la version d'exécution Amazon ECS qui inclut la version Docker.

Lorsque vous créez un cluster à l'aide de la console, Amazon ECS crée un modèle de lancement pour vos instances avec la dernière AMI associée au système d'exploitation sélectionné.

Lorsque vous créez AWS CloudFormation un cluster, le paramètre SSM fait partie du modèle de EC2 lancement Amazon pour les instances du groupe Auto Scaling. Vous pouvez configurer le modèle pour utiliser un paramètre dynamique de Systems Manager afin de déterminer l'AMI optimisée Amazon ECS à déployer. Ce paramètre garantit que chaque fois que vous déployez la pile, elle vérifie si une mise à jour disponible doit être appliquée aux EC2 instances. Pour un exemple d'utilisation du paramètre Systems Manager, consultez la section Créer un cluster Amazon ECS avec l'AMI Amazon Linux 2023 optimisée pour Amazon ECS dans le guide de l'AWS CloudFormation utilisateur.

L'ID de l'AMI, le nom de l'image, le système d'exploitation, la version de l'agent de conteneur, le nom de l'image source et la version d'exécution de chaque variante d'Amazon ECS-Optimized AMIs peuvent être récupérés par programmation en interrogeant l'API Systems Manager Parameter Store. Pour plus d'informations sur l'API Systems Manager Parameter Store, reportez-vous aux sections GetParameterset GetParametersByPath.

> **Note**
>
> Votre compte administratif doit avoir les autorisations IAM suivantes pour extraire les métadonnées d'AMI optimisée pour Amazon ECS. Ces autorisations ont été ajoutées à la politique IAM `AmazonECS_FullAccess`.
>
> - SMS : GetParameters
> - SMS : GetParameter
> - SMS : GetParametersByPath

Format de paramètre Systems Manager Parameter Store

Les informations ci-dessous présentent le format de nom de paramètre pour chaque variante d'AMI optimisée pour Amazon ECS.

Optimisé pour Linux Amazon ECS AMIs

- Métadonnées d'AMI Amazon Linux 2023 :

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/<version>
```

- Métadonnées d'AMI Amazon Linux 2023 (arm64) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/arm64/<version>
```

- Métadonnées d'AMI Amazon Linux 2023 (Neuron) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/neuron/<version>
```

- Métadonnées de l'AMI Amazon Linux 2023 (GPU) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/gpu/<version>
```

Métadonnées d'AMI Amazon Linux 2 :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/<version>
```

- Métadonnées d'AMI Amazon Linux 2 noyau 5.10 :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/<version>
```

- Métadonnées d'AMI Amazon Linux 2 (arm64) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/arm64/<version>
```

- Métadonnées d'AMI Amazon Linux 2 noyau 5.10 (arm64) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/arm64/<version>
```

- Métadonnées AMI du noyau 5.10 optimisées pour le GPU Amazon ECS :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/gpu/<version>
```

- Métadonnées d'AMI Amazon Linux 2 (GPU) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/<version>
```

- Métadonnées de l'AMI Amazon Linux 2 (Neuron) 5.10 optimisées pour Amazon ECS :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/inf/<version>
```

- Métadonnées d'AMI Amazon Linux 2 (Neuron) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/inf/<version>
```

Le format de nom de paramètre suivant permet de récupérer l'ID d'image de la dernière AMI Amazon Linux 2 optimisée pour Amazon ECS recommandée à l'aide du sous-paramètre. `image_id`

```
/aws/service/ecs/optimized-ami/amazon-linux-2/recommended/image_id
```

Le format de nom de paramètre suivant extrait les métadonnées d'une version spécifique d'AMI optimisée pour Amazon ECS en spécifiant le nom d'AMI.

- Métadonnées d'AMI Amazon Linux 2 optimisée pour Amazon ECS :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/amzn2-ami-ecs-hvm-2.0.20181112-x86_64-
ebs
```

> **Note**
>
> Toutes les versions de l'AMI Amazon Linux 2 optimisée pour Amazon ECS sont disponibles pour l'extraction. Seule l'AMI optimisée pour Amazon ECS versions `amzn-ami-2017.09.l-amazon-ecs-optimized` (Linux) et versions ultérieures peuvent être extraites.

Exemples

Les exemples suivants montrent comment vous pouvez récupérer les métadonnées de chaque variante d'AMI optimisée pour Amazon ECS.

Récupération des métadonnées de l'AMI optimisée pour Amazon ECS la plus récente recommandée

Vous pouvez récupérer la dernière AMI optimisée pour Amazon ECS recommandée à l' AWS CLI aide des commandes suivantes AWS CLI .

Optimisé pour Linux Amazon ECS AMIs

- Pour Amazon Linux 2023 optimisé pour Amazon ECS : AMIs

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended --region us-east-1
```

- Pour Amazon Linux 2023 (arm64) optimisé pour Amazon ECS : AMIs

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
arm64/recommended --region us-east-1
```

- Pour Amazon Linux 2023 (Neuron) optimisé pour Amazon ECS : AMIs

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
neuron/recommended --region us-east-1
```

- Pour le GPU Amazon Linux 2023 optimisé pour Amazon ECS : AMIs

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/gpu/
recommended --region us-east-1
```

- Pour le noyau Amazon Linux 2 5.10 optimisé pour Amazon ECS : AMIs

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/recommended --region us-east-1
```

- Pour Amazon Linux 2 optimisé pour Amazon ECS : AMIs

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
recommended --region us-east-1
```

- Pour le noyau Amazon Linux 2 5.10 (arm64) optimisé pour Amazon ECS : AMIs

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/arm64/recommended --region us-east-1
```

- Pour Amazon Linux 2 (arm64) optimisé pour Amazon ECS : AMIs

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/arm64/
recommended --region us-east-1
```

- Pour le noyau 5.10 optimisé pour le GPU Amazon ECS : AMIs

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/gpu/recommended --region us-east-1
```

- Pour l'Amazon ECS optimisé pour le GPU AMIs :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

- Pour le noyau Amazon Linux 2 (Neuron) AMIs 5.10 optimisé pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/inf/recommended --region us-east-1
```

- Pour Amazon Linux 2 (Neuron) AMIs optimisé pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/inf/
recommended --region us-east-1
```

Extraction de l'ID d'image de la dernière AMI Amazon Linux 2023 optimisée pour Amazon ECS recommandée

Vous pouvez extraire l'ID d'image de l'ID de la dernière AMI Amazon Linux 2023 optimisée pour Amazon ECS recommandée en utilisant le sous-paramètre `image_id`.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-
linux-2023/recommended/image_id --region us-east-1
```

Pour extraire uniquement la valeur `image_id`, vous pouvez interroger la valeur de paramètre spécifique ; par exemple :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended/image_id --region us-east-1 --query "Parameters[0].Value"
```

Extraction des métadonnées d'une version spécifique d'AMI Amazon Linux 2 optimisée pour Amazon ECS

Récupérez les métadonnées d'une version spécifique de l'AMI Amazon Linux optimisée pour Amazon ECS à l' AWS CLI aide de la commande suivante AWS CLI . Remplacez le nom d'AMI par le nom d'AMI Amazon Linux optimisée pour Amazon ECS pour procéder à l'extraction.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/amzn2-ami-ecs-hvm-2.0.20200928-x86_64-ebs --region us-east-1
```

Récupération des métadonnées AMI du noyau Amazon Linux 2 5.10 optimisées pour Amazon ECS à l'aide de l'API Systems Manager GetParametersByPath

Récupérez les métadonnées de l'AMI Amazon Linux 2 optimisées pour Amazon ECS avec l' GetParametersByPath API Systems Manager à l'aide de la AWS CLI commande suivante.

```
aws ssm get-parameters-by-path --path /aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/ --region us-east-1
```

Récupération de l'ID d'image de la dernière AMI Amazon Linux 2 5.10 recommandée optimisée pour Amazon ECS

Vous pouvez récupérer l'ID d'image du dernier ID AMI Amazon Linux 2 5.10 recommandé optimisé pour Amazon ECS en utilisant le sous-paramètre. `image_id`

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/recommended/image_id --region us-east-1
```

Pour extraire uniquement la valeur `image_id`, vous pouvez interroger la valeur de paramètre spécifique ; par exemple :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/recommended/image_id --region us-east-1 --query "Parameters[0].Value"
```

Utilisation de l'AMI optimisée pour Amazon ECS la plus récente recommandée dans un modèle AWS CloudFormation

Vous pouvez référencer la dernière AMI optimisée pour Amazon ECS recommandée dans un modèle AWS CloudFormation en référençant le nom du magasin de paramètres Systems Manager.

Exemple Linux

```
Parameters:kernel-5.10
  LatestECSOptimizedAMI:
    Description: AMI ID
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/recommended/
image_id
```

Migration d'un Amazon Linux 2 vers une AMI optimisée pour Amazon ECS pour Amazon Linux 2023

À la suite d'Amazon Linux, Amazon ECS met fin au support standard pour Amazon Linux 2 optimisé pour Amazon ECS à AMIs compter du 30 juin 2026. Après cette date, la version de l'agent Amazon ECS est épinglée et les nouveaux Amazon Linux 2 optimisés pour Amazon ECS ne AMIs sont publiés que lorsque l'AMI Amazon Linux 2 source est mise à jour. La fin de vie complète (EOL) a lieu le 30 juin 2026, après quoi aucun autre Amazon Linux AMIs 2 optimisé pour Amazon ECS n'est publié, même si l'AMI source est mise à jour.

Amazon Linux 2023 propose une secure-by-default approche avec des politiques de sécurité préconfigurées, SELinux en mode permissif, le mode IMDSv2 uniquement activé par défaut, des temps de démarrage optimisés et une gestion des packages améliorée pour une sécurité et des performances accrues.

Il existe un haut degré de compatibilité entre Amazon Linux 2 et Amazon Linux 2023 optimisé pour Amazon ECS AMIs, et la plupart des clients constateront des minimal-to-zero modifications de leur charge de travail entre les deux systèmes d'exploitation.

Pour plus d'informations, consultez la section Comparaison entre Amazon Linux 2 et Amazon Linux 2023 dans le guide de l'utilisateur Amazon Linux 2023 et dans le AL2023 FAQs.

Considérations de compatibilité

Gestion des packages et mises à jour du système d'exploitation

Contrairement aux versions précédentes d'Amazon Linux, Amazon Linux 2023 AMIs optimisé pour Amazon ECS est verrouillé sur une version spécifique du référentiel Amazon Linux. Cela permet aux utilisateurs de ne pas mettre à jour par inadvertance des packages susceptibles d'apporter des modifications indésirables ou irréversibles. Pour plus d'informations, consultez la section Gestion des référentiels et des mises à jour du système d'exploitation dans Amazon Linux 2023 dans le guide de l'utilisateur Amazon Linux 2023.

Versions du noyau Linux

Amazon Linux 2 est basé sur AMIs les noyaux Linux 4.14 et 5.10, tandis qu'Amazon Linux 2023 utilise les noyaux Linux 6.1 et 6.12. Pour plus d'informations, consultez la section Comparaison des noyaux Amazon Linux 2 et Amazon Linux 2023 dans le guide de l'utilisateur Amazon Linux 2023.

Modifications relatives à la disponibilité des packages

Les modifications de package les plus importantes apportées à Amazon Linux 2023 sont les suivantes :

- Certains packages binaires source d'Amazon Linux 2 ne sont plus disponibles dans Amazon Linux 2023. Pour plus d'informations, consultez la section Packages supprimés d'Amazon Linux 2023 dans les notes de mise à jour d'Amazon Linux 2023.
- Modifications apportées à la manière dont Amazon Linux prend en charge les différentes versions des packages. Le `amazon-linux-extras` système utilisé dans Amazon Linux 2 n'existe pas dans Amazon Linux 2023. Tous les packages sont simplement disponibles dans le référentiel « principal ».
- Les packages supplémentaires pour Enterprise Linux (EPEL) ne sont pas pris en charge dans Amazon Linux 2023. Pour plus d'informations, consultez la section Compatibilité EPEL dans Amazon Linux 2023 dans le guide de l'utilisateur Amazon Linux 2023.
- Les applications 32 bits ne sont pas prises en charge dans Amazon Linux 2023. Pour plus d'informations, consultez la section Fonctionnalités obsolètes d'Amazon Linux 2 dans le guide de l'utilisateur Amazon Linux 2023.

Modifications des groupes de contrôle (cgroups)

Un groupe de contrôle (cgroup) est une fonctionnalité du noyau Linux permettant d'organiser hiérarchiquement les processus et de répartir les ressources système entre eux. Les groupes de contrôle sont très fréquemment utilisés pour implémenter un environnement d'exécution de conteneur, et par `systemd`.

L'agent Amazon ECS, Docker et containerd prennent tous en charge à la fois cgroupv1 et cgroupv2. L'agent Amazon ECS et le moteur d'exécution du conteneur gèrent les cgroups pour vous, de sorte que les clients Amazon ECS n'ont pas besoin d'apporter de modifications pour cette mise à niveau de cgroup sous-jacente.

Pour plus de détails sur cgroupv2, consultez la section Groupes de contrôle v2 dans Amazon Linux 2023 dans le guide de l'utilisateur Amazon Linux 2023.

Modifications apportées au service de métadonnées d'instance (IMDS)

Amazon Linux 2023 nécessite la version 2 (IMDSv2) du service de métadonnées d'instance par défaut. IMDSv2 présente plusieurs avantages qui contribuent à améliorer la posture de sécurité. Il utilise une méthode d'authentification orientée session qui nécessite la création d'un jeton secret dans une simple requête HTTP PUT pour démarrer la session. Le jeton d'une session peut être valide entre 1 seconde et 6 heures.

Pour plus d'informations sur la manière de passer de IMDSv1 à IMDSv2, consultez la section Transition vers l'utilisation de la version 2 du service de métadonnées d'instance dans le guide de EC2 l'utilisateur Amazon.

Si vous souhaitez l'utiliser IMDSv1, vous pouvez toujours le faire en remplaçant manuellement les paramètres à l'aide des propriétés de lancement de l'option de métadonnées de l'instance.

Changements liés à l'échange de mémoire

L'échange de mémoire par conteneur n'est pas pris en charge sur Amazon Linux 2023 et cgroups v2. Pour de plus amples informations, veuillez consulter Gestion de l'espace mémoire d'échange de conteneurs sur Amazon ECS.

Modifications apportées à la validation FIPS

Amazon Linux 2 est certifié selon la norme FIPS 140-2 et Amazon Linux 2023 est certifié selon la norme FIPS 140-3.

Pour activer le mode FIPS sur Amazon Linux 2023, installez les packages nécessaires sur votre EC2 instance Amazon et suivez les étapes de configuration en suivant les instructions de la section Activer le mode FIPS sur Amazon Linux 2023 du guide de l'utilisateur Amazon Linux 2023.

Support d'instance accéléré

Amazon Linux 2023 optimisé pour Amazon ECS AMIs prend en charge les types d'instances accélérées Neuron et GPU. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.

Construire sur mesure AMIs

Bien que nous vous recommandions de passer à la version officiellement prise en charge et publiée par AMIs Amazon ECS optimisée pour Amazon Linux 2023, vous pouvez continuer à créer des

versions personnalisées d'Amazon Linux 2 optimisées pour Amazon ECS à AMIs l'aide des scripts de génération open source utilisés pour créer les variantes Linux de l'AMI optimisée pour Amazon ECS. Pour de plus amples informations, veuillez consulter Script de création d'AMI Linux optimisées pour Amazon ECS.

Stratégies de migration

Nous vous recommandons de créer et de mettre en œuvre un plan de migration incluant des tests approfondis des applications. Les sections suivantes décrivent les différentes stratégies de migration en fonction de la façon dont vous gérez votre infrastructure Amazon ECS.

Migration avec les fournisseurs de capacité Amazon ECS

1. Créez un nouveau fournisseur de capacité avec un nouveau modèle de lancement. Cela doit faire référence à un groupe Auto Scaling avec un modèle de lancement similaire à votre modèle existant, mais au lieu de l'AMI Amazon Linux 2 optimisée pour Amazon ECS, il doit spécifier l'une des variantes d'Amazon Linux 2023. Ajoutez ce nouveau fournisseur de capacité à votre cluster Amazon ECS existant.
2. Mettez à jour la stratégie de fournisseur de capacité par défaut de votre cluster pour inclure à la fois le fournisseur de capacité Amazon Linux 2 existant et le nouveau fournisseur de capacité Amazon Linux 2023. Commencez par un poids plus élevé chez le fournisseur Amazon Linux 2 et un poids inférieur chez le fournisseur Amazon Linux 2023 (par exemple, Amazon Linux 2 : poids 80, Amazon Linux 2023 : poids 20). Amazon ECS commence alors à approvisionner les instances Amazon Linux 2023 au fur et à mesure que de nouvelles tâches sont planifiées. Vérifiez que les instances s'enregistrent correctement et que les tâches peuvent s'exécuter correctement sur les nouvelles instances.
3. Ajustez progressivement le poids du fournisseur de capacité dans la stratégie par défaut de votre cluster, en augmentant le poids du fournisseur Amazon Linux 2023 tout en diminuant le poids du fournisseur Amazon Linux 2 au fil du temps (par exemple, 60/40, puis 40/60, puis 20/80). Vous pouvez également mettre à jour les stratégies de chaque fournisseur de capacité de service afin de prioriser les instances Amazon Linux 2023. Surveillez le placement des tâches pour vous assurer qu'elles s'exécutent correctement sur les instances Amazon Linux 2023.
4. Videz éventuellement les instances de conteneur Amazon Linux 2 pour accélérer la migration des tâches. Si vous disposez d'une capacité de remplacement suffisante pour Amazon Linux 2023, vous pouvez vider manuellement vos instances de conteneur Amazon Linux 2 via la console Amazon ECS ou AWS CLI pour accélérer la transition de vos tâches d'Amazon Linux 2 vers Amazon Linux 2023. Une fois la migration terminée, supprimez le fournisseur de capacité Amazon Linux 2 de votre cluster et supprimez le groupe Auto Scaling associé.

Migration avec un groupe Amazon EC2 Auto Scaling

1. Créez un nouveau groupe Amazon EC2 Auto Scaling avec un nouveau modèle de lancement. Il doit être similaire à votre modèle de lancement existant, mais au lieu de l'AMI Amazon Linux 2 optimisée pour Amazon ECS, il doit spécifier l'une des variantes d'Amazon Linux 2023. Ce nouveau groupe Auto Scaling peut lancer des instances sur votre cluster existant.
2. Élargissez le groupe Auto Scaling afin que les instances Amazon Linux 2023 commencent à s'enregistrer dans votre cluster. Vérifiez que les instances s'enregistrent correctement et que les tâches peuvent s'exécuter correctement sur les nouvelles instances.
3. Une fois que vos tâches ont été vérifiées pour fonctionner sur Amazon Linux 2023, augmentez le groupe Amazon Linux 2023 Auto Scaling tout en réduisant progressivement le groupe Amazon Linux 2 Auto Scaling, jusqu'à ce que vous ayez complètement remplacé toutes les instances Amazon Linux 2.
4. Si vous disposez d'une capacité de remplacement suffisante pour Amazon Linux 2023, vous souhaiterez peut-être vider explicitement les instances de conteneur afin d'accélérer la transition de vos tâches d'Amazon Linux 2 vers Amazon Linux 2023. Pour de plus amples informations, veuillez consulter Vidange des instances de conteneurs Amazon ECS.

Migration à l'aide d'instances gérées manuellement

1. Lancez manuellement (ou ajustez les scripts qui lancent) de nouvelles EC2 instances Amazon à l'aide de l'AMI Amazon Linux 2023 optimisée pour Amazon ECS au lieu d'Amazon Linux 2. Assurez-vous que ces instances utilisent les mêmes groupes de sécurité, sous-réseaux, rôles IAM et configuration de cluster que vos instances Amazon Linux 2 existantes. Les instances devraient s'enregistrer automatiquement dans votre cluster Amazon ECS existant lors du lancement.
2. Vérifiez que les nouvelles instances Amazon Linux 2023 sont correctement enregistrées dans votre cluster Amazon ECS et qu'elles sont en bon `ACTIVE` état. Vérifiez que les tâches peuvent être planifiées et exécutées correctement sur ces nouvelles instances en attendant le placement naturel des tâches ou en déclenchant manuellement la replanification de stopping/starting certaines tâches.
3. Remplacez progressivement vos instances Amazon Linux 2 en lançant des instances Amazon Linux 2023 supplémentaires selon les besoins, puis en vidant et en arrêtant manuellement les instances Amazon Linux 2 une par une. Vous pouvez vider les instances via la console Amazon ECS en définissant le `DRAINING` statut de l'instance, ce qui empêchera d'y placer de nouvelles tâches et permettra aux tâches existantes de se terminer ou d'être reprogrammées ailleurs.

Script de création d'AMI Linux optimisées pour Amazon ECS

Amazon ECS comporte des scripts de génération open source qui sont utilisés pour créer les variantes Linux de l'AMI optimisée pour Amazon ECS. Ces scripts de compilation sont désormais disponibles sur GitHub. Pour plus d'informations, voir amazon-ecs-amici-dessous GitHub.

Si vous devez personnaliser l'AMI optimisée pour Amazon ECS, consultez Amazon ECS Optimized AMI Build Recipies on. GitHub

Le référentiel de scripts de génération inclut un modèle de HashiCorppacker et des scripts de génération pour générer chacune des variantes Linux de l'AMI optimisée pour Amazon ECS. Ces scripts sont la source de vérité pour les versions d'AMI optimisées pour Amazon ECS. Vous pouvez donc suivre le GitHub référentiel pour suivre les modifications apportées à notre. AMIs Par exemple, vous pouvez souhaiter que votre propre AMI utilise la même version de Docker que celle utilisée par l'équipe Amazon ECS pour l'AMI officielle.

Pour plus d'informations, consultez le référentiel d'AMI Amazon ECS à l'adresse aws/ amazon-ecs-ami on GitHub.

Créer un AMI Linux optimisée pour Amazon ECS

1. Clonez le `aws/amazon-ecs-ami` GitHub dépôt.

   ```
   git clone https://github.com/aws/amazon-ecs-ami.git
   ```

2. Ajoutez une variable d'environnement à utiliser par la AWS région lors de la création de l'AMI. Remplacez la valeur `us-west-2` avec la Région à utiliser.

   ```
   export REGION=us-west-2
   ```

3. Un Makefile est fourni pour créer l'AMI. À partir du répertoire racine du référentiel cloné, utilisez l'une des commandes suivantes, correspondant à la variante Linux de l'AMI optimisée pour Amazon ECS que vous souhaitez créer.

   - AMI Amazon Linux 2 optimisée pour Amazon ECS

     ```
     make al2
     ```

   - AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS

```
make al2arm
```

- AMI optimisée pour GPU Amazon ECS

```
make al2gpu
```

- AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS

```
make al2inf
```

- AMI Amazon Linux 2023 optimisée pour Amazon ECS

```
make al2023
```

- AMI Amazon Linux 2023 (arm64) optimisée pour Amazon ECS

```
make al2023arm
```

- AMI Amazon Linux 2023 (Neuron) optimisée pour Amazon ECS

```
make al2023neu
```

Optimisé pour Amazon ECS Bottlerocket AMIs

Bottlerocketest un Linux système d'exploitation open source spécialement conçu AWS pour exécuter des conteneurs sur des machines virtuelles ou des hôtes bare metal. L'AMI Bottlerocket optimisée pour Amazon ECS est sécurisée et ne comprend que le nombre minimum de packages nécessaires à l'exécution des conteneurs. Cela améliore l'utilisation des ressources, réduit la surface d'attaque de sécurité et contribue à réduire les frais de gestion. L'AMI Bottlerocket est également intégrée à Amazon ECS afin de réduire la charge opérationnelle liée à la mise à jour des instances de conteneur dans un cluster.

Bottlerocket diffère d'Amazon Linux par les aspects suivants :

- Bottlerocket n'inclut pas de gestionnaire de packages et son logiciel ne peut être exécuté que sous forme de conteneurs. Les mises à jour de Bottlerocket sont appliquées et peuvent être annulées en une seule étape, ce qui réduit le risque d'erreurs de mise à jour.

- Le principal mécanisme de gestion des hôtes Bottlerocket est un planificateur de conteneurs. Contrairement à Amazon Linux, la connexion à des instances Bottlerocket individuelles est censée être une opération peu fréquente uniquement à des fins avancées de débogage et de dépannage.

Pour plus d'informations sur Bottlerocket, consultez la documentation et les versions sur GitHub.

Il existe des variantes de l'BottlerocketAMI optimisée pour Amazon ECS pour le noyau 6.1 et le noyau 5.10.

Les variantes suivantes utilisent le noyau 6.1 :

- `aws-ecs-2`
- `aws-ecs-2-nvidia`

Les variantes suivantes utilisent le noyau 5.10 :

- `aws-ecs-1`
- `aws-ecs-1-nvidia`

  Pour plus d'informations sur la variante `aws-ecs-1-nvidia`, veuillez consulter le billet de blog Announcing NVIDIA GPU support for Bottlerocket on Amazon ECS.

Considérations

Tenez compte des informations suivantes lors de l'utilisation d'une AMI Bottlerocket avec Amazon ECS.

- Bottlerocketprend en charge EC2 les instances Amazon avec `x86_64` et `arm64` les processeurs. Il n'est pas recommandé d'utiliser l'BottlerocketAMI avec des EC2 instances Amazon dotées d'une puce Inferentia.
- Les images Bottlerocket n'incluent pas un serveur SSH ou un shell. Cependant, vous pouvez utiliser out-of-band des outils de gestion pour obtenir un accès administrateur SSH et effectuer un amorçage. Pour plus d'informations, consultez ces sections dans le README.md de bottlerocket sur GitHub :
  - Exploration
  - Conteneur d'administration

- Par défaut, Bottlerocket possède un conteneur de contrôle qui est activé. Ce conteneur exécute l'AWS Systems Manager agent que vous pouvez utiliser pour exécuter des commandes ou démarrer des sessions shell sur des EC2 Bottlerocket instances Amazon. Pour plus d'informations, consultez Configuration du gestionnaire de session dans le Guide de l'utilisateur AWS Systems Manager .

- Bottlerocket est optimisé pour les applications de conteneurs et met l'accent sur la sécurité. Bottlerocket n'inclut pas de gestionnaire de package et est inaltérable. Pour plus d'informations sur les fonctionnalités de sécurité et les conseils de sécurité, voir Fonctionnalités de sécurité et conseils de sécurité sur GitHub.
- Le mode réseau `awsvpc` est pris en charge pour la version d'AMI Bottlerocket `1.1.0` ou version ultérieure.
- App Mesh dans une définition de tâche est pris en charge pour la version d'AMI Bottlerocket `1.15.0` ou ultérieure.
- Le paramètre de définition de `initProcessEnabled` tâche est pris en charge pour la `1.19.0` version Bottlerocket AMI ou ultérieure.
- Ils ne prennent pas Bottlerocket AMIs non plus en charge les services et fonctionnalités suivants :
  - ECS Anywhere
  - Service Connect
  - Amazon EFS en mode crypté
  - Amazon EFS en mode `awsvpc` réseau
  - Les volumes Amazon EBS ne peuvent pas être montés
  - Accélérateur Elastic Inference

Récupération des métadonnées d'AMI optimisées pour Amazon ECS Bottlerocket

Vous pouvez récupérer l'identifiant Amazon Machine Image (AMI) pour Amazon ECS-Optimized AMIs en interrogeant l'API AWS Systems Manager Parameter Store. Avec ce paramètre, vous n'avez pas besoin de rechercher manuellement l'AMI optimisée pour Amazon ECS. IDs Pour plus d'informations sur l'API du magasin de paramètres de Systems Manager, consultez GetParameter. L'utilisateur que vous utilisez doit disposer de l'autorisation IAM `ssm:GetParameter` pour récupérer les métadonnées de l'AMI optimisée pour Amazon ECS.

`aws-ecs-2`Variante Bottlerocket AMI

Vous pouvez récupérer la dernière variante stable de l'AMI `aws-ecs-2` Bottlerocket Région AWS en fonction de son architecture avec AWS CLI le ou le. AWS Management Console

- AWS CLI— Vous pouvez récupérer l'ID d'image de la dernière Bottlerocket AMI optimisée pour Amazon ECS recommandée à l'aide de la AWS CLI commande suivante en utilisant le sous-paramètre. `image_id` Remplacez la *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI. Pour plus d'informations sur les options prises en charge Régions AWS, consultez la section Trouver une AMI sur GitHub. Pour récupérer une autre version que la dernière, remplacez `latest` par le numéro de version.
  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    aws ssm get-parameter --region us-east-2 --name "/aws/service/bottlerocket/aws-
    ecs-2/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

    ```
    aws ssm get-parameter --region us-east-2 --name "/aws/service/bottlerocket/aws-
    ecs-2/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- AWS Management Console : vous pouvez interroger l'ID d'AMI optimisée pour Amazon ECS recommandée à l'aide d'une URL dans l' AWS Management Console. L'URL ouvre la console Amazon EC2 Systems Manager avec la valeur de l'ID du paramètre. Dans l'URL suivante, remplacez *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI. Pour plus d'informations sur les options prises en charge Régions AWS, consultez la section Trouver une AMI sur GitHub.
  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/
    aws-ecs-2/x86_64/latest/image_id/description?region=region#
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/
    aws-ecs-2/arm64/latest/image_id/description?region=region#
    ```

`aws-ecs-2-nvidia`Variante Bottlerocket AMI

Vous pouvez récupérer la dernière variante stable de l'AMI `aws-ecs-2-nvdia` Bottlerocket par région et architecture avec AWS CLI le ou le. AWS Management Console

- AWS CLI— Vous pouvez récupérer l'ID d'image de la dernière Bottlerocket AMI optimisée pour Amazon ECS recommandée à l'aide de la AWS CLI commande suivante en utilisant le sous-paramètre. `image_id` Remplacez la *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI. Pour plus d'informations sur les options prises en charge Régions AWS, consultez la section Trouver une AMI sur GitHub. Pour récupérer une autre version que la dernière, remplacez `latest` par le numéro de version.
  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-2-nvidia/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-2-nvidia/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- AWS Management Console : vous pouvez interroger l'ID d'AMI optimisée pour Amazon ECS recommandée à l'aide d'une URL dans l' AWS Management Console. L'URL ouvre la console Amazon EC2 Systems Manager avec la valeur de l'ID du paramètre. Dans l'URL suivante, remplacez *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI. Pour plus d'informations sur les options prises en charge Régions AWS, consultez la section Trouver une AMI sur GitHub.
  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    https://regionconsole.aws.amazon.com/systems-manager/parameters/aws/service/
    bottlerocket/aws-ecs-2-nvidia/x86_64/latest/image_id/description?region=region#
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

    ```
    https://regionconsole.aws.amazon.com/systems-manager/parameters/aws/service/
    bottlerocket/aws-ecs-2-nvidia/arm64/latest/image_id/description?region=region#
    ```

`aws-ecs-1`Variante Bottlerocket AMI

Vous pouvez récupérer la dernière variante stable de l'AMI `aws-ecs-1` Bottlerocket Région AWS en fonction de son architecture avec AWS CLI le ou le. AWS Management Console

- AWS CLI— Vous pouvez récupérer l'ID d'image de la dernière Bottlerocket AMI optimisée pour Amazon ECS recommandée à l'aide de la AWS CLI commande suivante en utilisant le sous-paramètre. `image_id` Remplacez la *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI. Pour plus d'informations sur les options prises en charge Régions AWS, consultez la section Trouver une AMI sur GitHub. Pour récupérer une autre version que la dernière, remplacez `latest` par le numéro de version.
  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-ecs-1/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-ecs-1/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- AWS Management Console : vous pouvez interroger l'ID d'AMI optimisée pour Amazon ECS recommandée à l'aide d'une URL dans l' AWS Management Console. L'URL ouvre la console Amazon EC2 Systems Manager avec la valeur de l'ID du paramètre. Dans l'URL suivante, remplacez *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI. Pour plus d'informations sur les options prises en charge Régions AWS, consultez la section Trouver une AMI sur GitHub.
  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    https://region.console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/aws-ecs-1/x86_64/latest/image_id/description
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

    ```
    https://region.console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/aws-ecs-1/arm64/latest/image_id/description
    ```

`aws-ecs-1-nvidia`Variante Bottlerocket AMI

Vous pouvez récupérer la dernière variante stable de l'AMI `aws-ecs-1-nvdia` Bottlerocket par région et architecture avec AWS CLI le ou le. AWS Management Console

- AWS CLI— Vous pouvez récupérer l'ID d'image de la dernière Bottlerocket AMI optimisée pour Amazon ECS recommandée à l'aide de la AWS CLI commande suivante en utilisant le sous-paramètre. `image_id` Remplacez la *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI. Pour plus d'informations sur les options prises en charge Régions AWS, consultez la section Trouver une AMI sur GitHub. Pour récupérer une autre version que la dernière, remplacez `latest` par le numéro de version.
  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1-nvidia/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1-nvidia/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- AWS Management Console : vous pouvez interroger l'ID d'AMI optimisée pour Amazon ECS recommandée à l'aide d'une URL dans l' AWS Management Console. L'URL ouvre la console Amazon EC2 Systems Manager avec la valeur de l'ID du paramètre. Dans l'URL suivante, remplacez *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI. Pour plus d'informations sur les options prises en charge Régions AWS, consultez la section Trouver une AMI sur GitHub.
  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/
    aws-ecs-1-nvidia/x86_64/latest/image_id/description?region=region#
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/
    aws-ecs-1-nvidia/arm64/latest/image_id/description?region=region#
    ```

Étapes suivantes

Pour un didacticiel détaillé sur la prise en main du système Bottlerocket d'exploitation sur Amazon ECS, consultez les sections Utilisation d'une AMI Bottlerocket avec Amazon ECS GitHub activé et Getting started with Bottlerocket et Amazon ECS sur AWS le site du blog.

Pour plus d'informations sur le lancement d'une instance Bottlerocket, voir Lancement d'une Bottlerocket instance pour Amazon ECS

Lancement d'une Bottlerocket instance pour Amazon ECS

Vous pouvez lancer une Bottlerocket instance afin de pouvoir exécuter vos charges de travail de conteneur.

Vous pouvez utiliser le AWS CLI pour lancer l'Bottlerocketinstance.

1. Créez un fichier appelé `userdata.toml`. Ce fichier est utilisé pour les données utilisateur de l'instance. Remplacez *`cluster-name`* par le nom de votre cluster.

```
[settings.ecs]
cluster = "cluster-name"
```

2. Utilisez l'une des commandes incluses dans the section called “Récupération des métadonnées d'AMI optimisées pour Amazon ECS Bottlerocket” pour obtenir l'ID d'AMI Bottlerocket. Utilisez ceci lors de l'étape suivante.
3. Exécutez la commande suivante pour lancer l'instance Bottlerocket. N'oubliez pas de remplacer les paramètres suivants :

   - *`subnet`*Remplacez-le par l'ID du sous-réseau privé ou public dans lequel votre instance sera lancée.
   - *`bottlerocket_ami`*Remplacez-le par l'ID AMI de l'étape précédente.
   - *`t3.large`*Remplacez-le par le type d'instance que vous souhaitez utiliser.
   - Remplacez *`region`* par le code de région.

```
aws ec2 run-instances --key-name ecs-bottlerocket-example \
   --subnet-id subnet \
   --image-id bottlerocket_ami \
   --instance-type t3.large \
   --region region \
```

```
  --tag-specifications
'ResourceType=instance,Tags=[{Key=bottlerocket,Value=example}]' \
  --user-data file://userdata.toml \
  --iam-instance-profile Name=ecsInstanceRole
```

4. Exécutez la commande suivante pour vérifier que l'instance de conteneur est enregistrée auprès du cluster. Lorsque vous exécutez cette commande, n'oubliez pas de remplacer les paramètres suivants :

   - Remplacez *cluster* par le nom de votre cluster.
   - *region*Remplacez-le par votre code de région.

```
aws ecs list-container-instances --cluster cluster-name --region region
```

Pour une présentation détaillée de la prise en main du système Bottlerocket d'exploitation sur Amazon ECS, consultez les sections Utilisation d'une AMI Bottlerocket avec Amazon ECS activé et Getting started with Bottlerocketet Amazon ECS GitHub sur le AWS site du blog.

Gestion des instances de conteneurs Linux Amazon ECS

Lorsque vous utilisez EC2 des instances pour vos charges de travail Amazon ECS, vous êtes responsable de leur maintenance

Procédures de gestion

- Lancement d'une instance de conteneur Amazon ECS Linux
- Démarrage des instances de conteneur Linux Amazon ECS pour transmettre des données
- Configuration des instances de conteneur Linux Amazon ECS pour recevoir des notifications relatives aux instances Spot
- Exécution d'un script lorsque vous lancez une instance de conteneur Linux Amazon ECS
- Augmenter les interfaces réseau des instances de conteneur Linux Amazon ECS
- Réservation de mémoire d'instance de conteneur Amazon ECS Linux
- Gestion à distance des instances de conteneur Amazon ECS à l'aide de AWS Systems Manager
- Utilisation d'un proxy HTTP pour les instances de conteneur Amazon ECS Linux
- Configuration d'instances pré-initialisées pour votre groupe Amazon ECS Auto Scaling
- Mise à jour de l'agent de conteneur Amazon ECS

Chaque version d'agent de conteneur Amazon ECS prend en charge un ensemble de différentes fonctions et fournit des correctifs des versions précédentes. Dans la mesure du possible, nous recommandons toujours d'utiliser la dernière version de l'agent de conteneur Amazon ECS. Pour mettre à jour votre agent de conteneur avec la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Pour voir quelles fonctionnalités et améliorations sont incluses dans chaque version de l'agent, consultez https://github.com/aws/amazon-ecs-agent/releases.

> **Important**
>
> La version minimale de Docker pour des métriques fiables est la version `v20.10.13` et les versions ultérieures, qui sont incluses dans l'AMI `20220607` optimisée pour Amazon ECS et les versions plus récentes.
> La version de l'agent Amazon ECS `1.20.0` et les versions plus récentes ne prennent plus en charge les versions de Docker antérieures à `18.01.0`.

Lancement d'une instance de conteneur Amazon ECS Linux

Vous pouvez créer des instances de conteneur Amazon ECS à l'aide de la EC2 console Amazon.

Vous pouvez lancer une instance par différentes méthodes AWS CLI, notamment la EC2 console Amazon et le SDK. Pour plus d'informations sur les autres méthodes de lancement d'une instance, consultez Lancer votre instance dans le guide de EC2 l'utilisateur Amazon.

Pour plus d'informations sur l'assistant de lancement, consultez Lancer une instance à l'aide du nouvel assistant de lancement d'instance dans le guide de EC2 l'utilisateur Amazon.

Avant de commencer, complétez les étapes détaillées dans Configurer l'utilisation d'Amazon ECS.

Vous pouvez utiliser le nouvel EC2 assistant Amazon pour lancer une instance. L’assistant de lancement d’instance spécifie les paramètres de lancement qui sont requis pour lancer une instance.

Paramètres pour la configuration d’instance

- Procédure
- Noms et identifications
- Images d’applications et de systèmes d’exploitation (Amazon Machine Image)

- Type d’instance
- Paire de clés (connexion)
- Paramètres réseau
- Configurer le stockage
- Détails avancés

Procédure

Avant de commencer, complétez les étapes détaillées dans Configurer l'utilisation d'Amazon ECS.

1. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans la barre de navigation en haut de l'écran, la AWS région actuelle est affichée (par exemple, USA East (Ohio)). Sélectionnez une région dans laquelle lancer l’instance.
3. Dans le tableau de bord de EC2 la console Amazon, choisissez Launch instance.

Noms et identifications

Le nom de l'instance est une identification, où la clé est Name (Nom), et la valeur est le nom que vous spécifiez. Vous pouvez étiqueter l'instance, les volumes et les Elastic Graphics. Pour les instances Spot, vous pouvez baliser uniquement la demande d’instance Spot.

La spécification d’un nom d’instance et d’identifications supplémentaires est facultative.

- Pour Name (Nom), saisissez un nom descriptif pour l’instance. Si vous ne spécifiez pas de nom, l’instance peut être identifiée par son ID, qui est automatiquement généré lorsque vous lancez l’instance.
- Pour ajouter des identifications supplémentaires, sélectionnez Add additional tags (Ajouter des identifications supplémentaires). Choisissez Add tag (Ajouter une identification), saisissez une clé et une valeur, puis sélectionnez le type de ressource à étiqueter. Choisissez Add tag (Ajouter une identification) pour chaque étiquette supplémentaire.

Images d’applications et de systèmes d’exploitation (Amazon Machine Image)

Une Amazon Machine Image (AMI) contient les informations requises pour créer une instance. Par exemple, une AMI peut contenir le logiciel nécessaire pour fonctionner en tant que serveur web, comme Apache et votre site web.

Utilisez la barre de recherche pour trouver une AMI optimisée pour Amazon ECS adaptée publiée par. AWS

1. Entrez l'un des termes suivants dans la barre de recherche.
   - **`ami-ecs`**
   - La valeur d'une AMI optimisée pour Amazon ECS.

     Pour connaître les dernières versions optimisées pour Amazon ECS AMIs et leurs valeurs, consultez l'AMI optimisée pour Amazon ECS pour Linux.
2. Appuyez sur Entrée.
3. Sur la page Choose an Amazon Machine Image (AMI), sélectionnez l' AMIsonglet AWS Marketplace.
4. À partir du volet de gauche Refine results (Affiner les résultats), sélectionnez Amazon Web Services en tant qu'éditeur.
5. Choisissez Sélectionner sur la ligne de l'AMI que vous souhaitez utiliser.

   Vous pouvez également choisir Cancel (Annuler) (en haut à droite) pour revenir à l'assistant de lancement d'instance sans choisir une AMI. Une AMI par défaut sera sélectionnée. Assurez-vous que l'AMI répond aux exigences décrites dans le système Linux optimisé pour Amazon ECS. AMIs

Type d’instance

Le type d’instance définit la configuration matérielle et la taille de l’instance. Les types d’instance plus importants disposent de plus d’UC et de mémoire. Pour plus d’informations, consultez Types d’instance.

- Pour Instance type (Type d’instance), sélectionnez le type de l’instance.

  Le type d'instance que vous sélectionnez détermine les ressources disponibles pour les tâches à exécuter.

Paire de clés (connexion)

Pour Key pair name (Nom de la paire de clés), choisissez une paire de clés existante ou choisissez Create new key pair (Créer une nouvelle paire de clés) pour en créer une nouvelle.

> **Important**
>
> Si vous sélectionnez l'option Proceed without key pair (Not recommended) ((Continuer sans paire de clé) (Non recommandé)), vous ne pourrez pas vous connecter à l'instance à moins de choisir une AMI configurée de façon à autoriser les utilisateurs à se connecter d'une autre façon.

Paramètres réseau

Configurez les paramètres réseau, le cas échéant.

- Networking platform (Plateforme de mise en réseau) : choisissez Virtual Private Cloud (VPC) (Cloud privé virtuel [VPC]), puis spécifiez le sous-réseau dans la section Network interfaces (Interfaces réseau).
- VPC : sélectionnez un VPC existant dans lequel créer le groupe de sécurité.
- Sous-réseau : vous pouvez lancer une instance dans un sous-réseau associé à une zone de disponibilité, une zone locale, une zone Wavelength ou un Outpost.

  Pour lancer l'instance dans une zone de disponibilité, sélectionnez le sous-réseau dans lequel lancer votre instance. Pour créer un sous-réseau, choisissez Créer un nouveau sous-réseau afin d'accéder à la console Amazon VPC. Une fois que vous avez terminé, revenez dans l'assistant de lancement d'instance et choisissez l'icône Refresh (Actualiser) afin de charger votre sous-réseau dans la liste.

  Pour lancer l'instance dans une zone locale, sélectionnez un sous-réseau que vous avez créé dans la zone locale.

  Pour lancer une instance dans un Outpost, sélectionnez un sous-réseau dans un VPC que vous avez associé à l'Outpost.
- Auto-assign Public IP (Attribuer automatiquement l'adresse IP publique) : si votre instance doit être accessible à partir d'Internet, vérifiez que le champ Auto-assign Public IP (Attribuer automatiquement l'adresse IP publique) est défini sur Enable (Activer). Si ce n'est pas le cas, définissez ce champ sur Disable (Désactiver).

> **Note**
>
> Les instances de conteneur ont besoin de communiquer avec le point de terminaison de service Amazon ECS. Cela peut être via un point de terminaison d'un VPC d'interface ou via vos instances de conteneur ayant des adresses IP publiques.
> Pour plus d'informations sur les points de terminaison d'un VPC d'interface, consultez Points de terminaison d'un VPC d'interface Amazon ECS (AWS PrivateLink).
> Si vous n'avez pas de point de terminaison d'un VPC d'interface configuré et que vos instances de conteneur n'ont pas d'adresses IP publiques, elles doivent utiliser la traduction d'adresse réseau (NAT) pour fournir cet accès. Pour de plus amples informations, veuillez consulter Passerelles NAT dans le Guide de l'utilisateur Amazon VPC et Utilisation d'un proxy HTTP pour les instances de conteneur Amazon ECS Linux dans ce guide.

- Firewall (security groups) (Pare-feu (groupes de sécurité)) : utilisez un groupe de sécurité afin de définir les règles de pare-feu de votre instance de conteneur. Ces règles déterminent le trafic réseau entrant acheminé vers votre instance de conteneur. Le reste du trafic est ignoré.
  - Pour sélectionner un groupe de sécurité existant, choisissez Select existing security group (Sélectionner un groupe de sécurité existant), puis sélectionnez le groupe de sécurité que vous avez créé dans Configurer l'utilisation d'Amazon ECS.

Configurer le stockage

L'AMI sélectionnée inclut un ou plusieurs volumes de stockage, notamment le volume racine. Vous pouvez spécifier d'autres volumes à attacher à l'instance.

Vous pouvez utiliser la vue Simple.

- Storage type (Type de stockage) : configurez le stockage pour votre instance de conteneur.

  Si vous utilisez l'AMI Amazon Linux 2 optimisée pour Amazon ECS, votre instance contient un seul volume configuré de 30 Gio, partagé entre le système d'exploitation et Docker.

  Si vous utilisez l'AMI optimisée pour Amazon ECS, votre instance contient deux volumes configurés. Le volume Racine est réservé au système d'exploitation et le second volume Amazon EBS (attaché à `/dev/xvdcz`) est réservé à Docker.

  Vous pouvez augmenter ou diminuer la taille des volumes pour votre instance afin de répondre aux besoins de votre application.

Détails avancés

Développez la section Détails avancés pour afficher les champs et spécifier des paramètres supplémentaires pour l'instance.

- Purchasing option (Option d'achat) : sélectionnez Request Spot Instances (Demander des instances Spot) pour demander une instance Spot. Vous devez également définir les autres champs associés aux instances Spot. Pour plus d'informations, consultez Demandes d'instances Spot.

  > **Note**
  >
  > Si vous utilisez les instances Spot et qu'un message `Not available` s'affiche, vous devrez peut-être choisir un autre type d'instance.

  .

- IAM instance profile (Profil d'instance IAM) : sélectionnez votre rôle IAM d'instance de conteneur. Celui-ci est généralement nommé `ecsInstanceRole`.

  > **Important**
  >
  > Si vous ne pas lancez votre instance de conteneur avec les autorisations IAM appropriées, votre agent Amazon ECS ne peut pas se connecter à votre cluster. Pour de plus amples informations, veuillez consulter Rôle IAM d'instance de conteneur Amazon ECS.

- Données utilisateur : configurez votre instance de conteneur Amazon ECS avec des données utilisateur, telles que les variables d'environnement de l'agent provenant deConfiguration de l'agent de conteneur Amazon ECS. Les scripts de données EC2 utilisateur Amazon ne sont exécutés qu'une seule fois, lors du premier lancement de l'instance. Les exemples suivants présentent des utilisations courantes des données utilisateur :
  - Par défaut, votre instance de conteneur est lancée dans votre cluster par défaut. Pour un lancement dans un cluster autre que celui défini par défaut, choisissez la liste Détails avancés. Collez ensuite le script suivant dans le champ Données utilisateur, en le *`your_cluster_name`* remplaçant par le nom de votre cluster.

    ```
    #!/bin/bash
    echo ECS_CLUSTER=your_cluster_name >> /etc/ecs/ecs.config
    ```

- Si vous avez un fichier `ecs.config` dans Amazon S3 et que l'accès en lecture seule d'Amazon S3 à votre rôle d'instance de conteneur est activé, choisissez la liste Advanced Details (Détails avancés). Collez ensuite le script suivant dans le champ Données utilisateur, en le *your_bucket_name* remplaçant par le nom de votre bucket pour installer le AWS CLI et écrivez votre fichier de configuration au moment du lancement.

> **Note**
>
> Pour en savoir plus sur cette configuration, consultez Stockage de la configuration de l'instance de conteneur Amazon ECS dans Amazon S3.

```
#!/bin/bash
yum install -y aws-cli
aws s3 cp s3://your_bucket_name/ecs.config /etc/ecs/ecs.config
```

- Spécifiez des balises pour votre instance de conteneur à l'aide du paramètre de configuration `ECS_CONTAINER_INSTANCE_TAGS`. Cela crée des balises associées uniquement à Amazon ECS, elles ne peuvent pas être répertoriées à l'aide de l' EC2 API Amazon.

> **Important**
>
> Si vous lancez vos instances de conteneur à l'aide d'un groupe Amazon EC2 Auto Scaling, vous devez utiliser le paramètre de configuration de l'agent ECS_CONTAINER_INSTANCE_TAGS pour ajouter des balises. Cela est dû à la manière dont les balises sont ajoutées aux EC2 instances Amazon lancées à l'aide de groupes Auto Scaling.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=your_cluster_name
ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
EOF
```

- Spécifiez des balises pour votre instance de conteneur, puis utilisez le paramètre de `ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM` configuration pour les propager d'Amazon EC2 vers Amazon ECS

L'exemple suivant présente un script de données utilisateur qui propage les balises associées à une instance de conteneur, et enregistre l'instance de conteneur avec un cluster nommé `your_cluster_name` :

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=your_cluster_name
ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM=ec2_instance
EOF
```

Pour de plus amples informations, veuillez consulter Démarrage des instances de conteneur Linux Amazon ECS pour transmettre des données.

Démarrage des instances de conteneur Linux Amazon ECS pour transmettre des données

Lorsque vous lancez une EC2 instance Amazon, vous pouvez lui transmettre des EC2 données utilisateur. Ces données peuvent être utilisées afin d'effectuer des tâches de configuration automatisées courantes, voire d'exécuter des scripts lors du démarrage de l'instance. Pour Amazon ECS, les données utilisateur sont le plus souvent utilisées pour transmettre les informations de configuration au démon Docker et à l'agent de conteneur Amazon ECS.

Vous pouvez transmettre plusieurs types de données utilisateur à Amazon EC2, notamment des boothooks cloud, des scripts shell et `cloud-init` des directives. Pour plus d'informations sur ce point et sur d'autres types de format, consultez la documentation sur Cloud-Init.

Pour transmettre les données utilisateur lors de l'utilisation de l'assistant de EC2 lancement Amazon, consultezLancement d'une instance de conteneur Amazon ECS Linux.

Vous pouvez configurer l'instance de conteneur pour transmettre des données dans la configuration de l'agent de conteneur ou dans la configuration du daemon Docker.

Agent de conteneur Amazon ECS

Les variantes Linux de l'AMI optimisée pour Amazon ECS recherchent les données de configuration de l'agent dans le fichier `/etc/ecs/ecs.config` au démarrage de l'agent de conteneur. Vous pouvez spécifier ces données de configuration au lancement avec les données EC2 utilisateur Amazon. Pour plus d'informations sur les variables de configuration d'agent de conteneur Amazon ECS disponibles, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

Pour définir une seule variable de configuration d'agent, par exemple le nom du cluster, utilisez echo afin de copier la variable dans le fichier de configuration :

```
#!/bin/bash
echo "ECS_CLUSTER=MyCluster" >> /etc/ecs/ecs.config
```

Si vous devez écrire plusieurs variables dans `/etc/ecs/ecs.config`, utilisez le format `heredoc` suivant. Ce format écrit tout entre les lignes commençant par cat et `EOF` dans le fichier de configuration.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_ENGINE_AUTH_TYPE=docker
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"username":"my_name","password":"my_password","email":"email@example.com"}}
ECS_LOGLEVEL=debug
ECS_WARM_POOLS_CHECK=true
EOF
```

Pour définir des attributs d'instance personnalisés, définissez la variable d'environnement `ECS_INSTANCE_ATTRIBUTES`.

```
#!/bin/bash
cat <<'EOF' >> ecs.config
ECS_INSTANCE_ATTRIBUTES={"envtype":"prod"}
EOF
```

Démon Docker

Vous pouvez spécifier les informations de configuration du démon Docker à l'aide des données EC2 utilisateur Amazon. Pour plus d'informations sur les options de configuration , consultez la documentation sur le démon Docker.

> **Note**
>
> AWS ne prend pas en charge les configurations Docker personnalisées, car elles peuvent parfois entrer en conflit avec les futures modifications ou fonctionnalités d'Amazon ECS sans avertissement.

Dans l'exemple ci-dessous, les options personnalisées sont ajoutées au fichier de configuration du démon Docker, `/etc/docker/daemon.json`, qui est ensuite spécifié dans les données utilisateur lors du lancement de l'instance.

```
#!/bin/bash
cat <<EOF >/etc/docker/daemon.json
{"debug": true}
EOF
systemctl restart docker --no-block
```

Dans l'exemple ci-dessous, les options personnalisées sont ajoutées au fichier de configuration du démon Docker, `/etc/docker/daemon.json`, qui est ensuite spécifié dans les données utilisateur lors du lancement de l'instance. Cet exemple montre comment désactiver le proxy Docker dans le fichier de configuration du démon Docker.

```
#!/bin/bash
cat <<EOF >/etc/docker/daemon.json
{"userland-proxy": false}
EOF
systemctl restart docker --no-block
```

Configuration des instances de conteneur Linux Amazon ECS pour recevoir des notifications relatives aux instances Spot

Amazon EC2 met fin, arrête ou met en veille prolongée votre instance Spot lorsque le prix Spot dépasse le prix maximum pour votre demande ou lorsque la capacité n'est plus disponible. Amazon EC2 fournit un préavis d'interruption de deux minutes à l'instance Spot pour les actions de résiliation et d'arrêt. Il ne fournit pas l'avis de deux minutes pour l'action de mise en veille prolongée. Si le drainage des instances Spot Amazon ECS est activé sur l'instance, Amazon ECS reçoit l'avis d'interruption de l'instance Spot et place l'instance en `DRAINING` statut.

> **Important**
>
> Amazon ECS ne reçoit aucune notification d'Amazon EC2 lorsque des instances sont supprimées par Auto Scaling Capacity Rebalancing. Pour plus d'informations, consultez Amazon EC2 Auto Scaling Capacity Rebalancing.

Lorsqu'une instance de conteneur est définie sur `DRAINING`, Amazon ECS bloque la planification du placement des nouvelles tâches sur l'instance de conteneur. Les tâches de service ayant l'état `PENDING` sur l'instance de conteneur faisant l'objet du drainage sont arrêtées immédiatement. S'il y a des instances de conteneur disponibles dans le cluster, des tâches de service de remplacement sont lancées dessus.

Le drainage des instances Spot est désactivé par défaut.

Vous pouvez activer le drainage des instances Spot lorsque vous lancez une instance. Ajoutez le script suivant dans le champ Données utilisateur. Remplacez *`MyCluster`* par le nom du cluster dans lequel enregistrer l'instance de conteneur.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_ENABLE_SPOT_INSTANCE_DRAINING=true
EOF
```

Pour de plus amples informations, veuillez consulter Lancement d'une instance de conteneur Amazon ECS Linux.

Pour activer le drainage des instances Spot pour une instance de conteneur existante

1. Connectez-vous à l'instance Spot via SSH.
2. Modifiez le fichier `/etc/ecs/ecs.config` et ajoutez le code suivant :

   ```
   ECS_ENABLE_SPOT_INSTANCE_DRAINING=true
   ```

3. Redémarrez le service `ecs`.

   - Pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS :

     ```
     sudo systemctl restart ecs
     ```

4. (Facultatif) Vous pouvez vérifier que l'agent est en cours d'exécution et consulter des informations sur votre nouvelle instance de conteneur en interrogeant l'opération API d'introspection d'agent. Pour de plus amples informations, veuillez consulter the section called “Introspection des conteneurs”.

   ```
   curl http://localhost:51678/v1/metadata
   ```

Exécution d'un script lorsque vous lancez une instance de conteneur Linux Amazon ECS

Vous devrez peut-être exécuter un conteneur spécifique sur chaque instance de conteneur pour gérer les opérations ou les problèmes de sécurité tels que la surveillance, la sécurité, les métriques, la découverte de services ou la journalisation.

Pour ce faire, vous pouvez configurer vos instances de conteneur de façon à ce qu'elles appellent la commande docker run avec le script de données utilisateur au moment du lancement ou dans un système d'initialisation comme Upstart ou systemd. Bien que cette méthode soit efficace, elle présente quelques inconvénients, étant donné qu'Amazon ECS n'a pas connaissance du conteneur et qu'il ne peut pas surveiller l'UC, la mémoire, les ports ni aucune autre ressource utilisée. Afin de veiller à ce qu'Amazon ECS prenne en compte correctement toutes les ressources de la tâche, vous devez créer une définition de tâche pour le conteneur que vous voulez exécuter sur vos instances de conteneur. Utilisez ensuite Amazon ECS pour placer la tâche avec les données EC2 utilisateur Amazon au moment du lancement.

Le script de données EC2 utilisateur Amazon décrit dans la procédure suivante utilise l'API d'introspection Amazon ECS pour identifier l'instance de conteneur. Ensuite, il utilise la start-task commande AWS CLI et pour exécuter une tâche spécifiée sur lui-même au démarrage.

Pour démarrer une tâche au moment du lancement d'une instance de conteneur

1. Modifiez votre rôle IAM `ecsInstanceRole` pour ajouter des autorisations pour l'opération d'API `StartTask`. Pour plus d'informations, consultez la section Mettre à jour les autorisations pour un rôle dans le Guide de AWS Identity and Access Management l'utilisateur.
2. Lancez une ou plusieurs instances de conteneur à l'aide de l'AMI Amazon Linux 2 optimisée pour Amazon ECS. Lancez de nouvelles instances de conteneur et utilisez l'exemple de script suivant dans les données EC2 utilisateur. *`your_cluster_name`*Remplacez-le par le cluster dans lequel l'instance de conteneur doit être enregistrée et *`my_task_def`* par la définition de tâche à exécuter sur l'instance au lancement.

   Pour de plus amples informations, veuillez consulter Lancement d'une instance de conteneur Amazon ECS Linux.

   > **Note**
   >
   > Le contenu du fichier MIME en plusieurs parties ci-dessous utilise un script shell pour définir les valeurs de configuration et installer les packages. Il utilise également une

tâche systemd pour démarrer la tâche après le lancement du service ecs et une fois que l'API d'introspection est disponible.

```
Content-Type: multipart/mixed; boundary="==BOUNDARY==" 
MIME-Version: 1.0

--==BOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
# Specify the cluster that the container instance should register into
cluster=your_cluster_name

# Write the cluster configuration variable to the ecs.config file
# (add any other configuration variables here also)
echo ECS_CLUSTER=$cluster >> /etc/ecs/ecs.config

START_TASK_SCRIPT_FILE="/etc/ecs/ecs-start-task.sh"
cat <<- 'EOF' > ${START_TASK_SCRIPT_FILE}
 exec 2>>/var/log/ecs/ecs-start-task.log
 set -x

 # Install prerequisite tools
 yum install -y jq aws-cli

 # Wait for the ECS service to be responsive
 until curl -s http://localhost:51678/v1/metadata
 do
  sleep 1
 done

 # Grab the container instance ARN and AWS Region from instance metadata
 instance_arn=$(curl -s http://localhost:51678/v1/metadata | jq -r '.
 | .ContainerInstanceArn' | awk -F/ '{print $NF}' )
 cluster=$(curl -s http://localhost:51678/v1/metadata | jq -r '. | .Cluster' | awk
 -F/ '{print $NF}' )
 region=$(curl -s http://localhost:51678/v1/metadata | jq -r '.
 | .ContainerInstanceArn' | awk -F: '{print $4}')

 # Specify the task definition to run at launch
 task_definition=my_task_def
```

```
 # Run the AWS CLI start-task command to start your task on this container instance
 aws ecs start-task --cluster $cluster --task-definition $task_definition --
container-instances $instance_arn --started-by $instance_arn --region $region
EOF

# Write systemd unit file
UNIT="ecs-start-task.service"
cat <<- EOF > /etc/systemd/system/${UNIT}
      [Unit]
      Description=ECS Start Task
      Requires=ecs.service
      After=ecs.service

      [Service]
      Restart=on-failure
      RestartSec=30
      ExecStart=/usr/bin/bash ${START_TASK_SCRIPT_FILE}

      [Install]
      WantedBy=default.target
EOF

# Enable our ecs.service dependent service with `--no-block` to prevent systemd
 deadlock
# See https://github.com/aws/amazon-ecs-agent/issues/1707
systemctl enable --now --no-block "${UNIT}"
--==BOUNDARY==--
```

3. Vérifiez que vos instances de conteneur sont lancées dans le cluster approprié et que vos tâches ont démarré.

   a. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.

   b. Dans la barre de navigation, sélectionnez la région dans laquelle se trouve votre cluster.

   c. Dans le panneau de navigation, choisissez Clusters, puis sélectionnez le cluster qui héberge vos instances de conteneur.

   d. Sur la page Cluster, choisissez Tâches, puis choisissez vos tâches.

      Chaque instance de conteneur que vous avez lancée doit comporter votre tâche en cours d'exécution.

Si vous ne voyez pas vos tâches, vous pouvez vous connecter à vos instances de conteneur avec le protocole SSH et vérifier le fichier `/var/log/ecs/ecs-start-task.log` pour consulter les informations de débogage.

Augmenter les interfaces réseau des instances de conteneur Linux Amazon ECS

> **Note**
>
> Cette fonction n'est pas disponible sur Fargate.

Chaque tâche qui utilise le mode `awsvpc` réseau reçoit sa propre interface Elastic Network (ENI), qui est attachée à l'instance de conteneur qui l'héberge. Le nombre d'interfaces réseau pouvant être associées à une EC2 instance Amazon est limité par défaut, et l'interface réseau principale compte pour une. Par exemple, par défaut, jusqu'à trois `c5.large` instances peuvent être ENIs associées. L'interface réseau principale de l'instance compte pour une seule, vous pouvez donc en attacher deux autres ENIs à l'instance. Comme chaque tâche utilisant le mode `awsvpc` réseau nécessite unENI, vous ne pouvez généralement exécuter que deux tâches de ce type sur ce type d'instance.

Amazon ECS prend en charge le lancement d'instances de conteneur avec une ENI densité accrue à l'aide des types d' EC2 instances Amazon pris en charge. Lorsque vous utilisez ces types d'instances et que vous activez le paramètre du `awsvpcTrunking` compte, d'autres ENIs sont disponibles sur les instances de conteneur récemment lancées. Cette configuration vous permet de placer plus de tâches sur chaque instance de conteneur. Pour utiliser la console afin d'activer la fonctionnalité, voirModification des paramètres du compte Amazon ECS. Pour utiliser le AWS CLI pour activer la fonctionnalité, voirGestion des paramètres du compte Amazon ECS à l'aide du AWS CLI.

Par exemple, une `c5.large` instance avec `awsvpcTrunking` une ENI limite accrue de douze. L'instance de conteneur a alors une interface réseau principale. Amazon ECS crée et attache une interface réseau « tronc » à l'instance de conteneur. Par conséquent, cette configuration vous permet de lancer dix tâches sur l'instance de conteneur au lieu des deux tâches actuellement possibles.

L'interface réseau « tronc » est entièrement gérée par Amazon ECS et est supprimée lorsque vous résiliez votre instance de conteneur ou lorsque vous annulez son enregistrement dans le cluster. Pour de plus amples informations, veuillez consulter Options de mise en réseau des tâches Amazon ECS pour le type de EC2 lancement.

Considérations

Tenez compte des points suivants lorsque vous utilisez la fonction de ENI jonction.

- Seules les variantes Linux de l'AMI optimisée pour Amazon ECS, ou les autres variantes Amazon Linux avec une version `1.28.1` ou une version ultérieure de l'agent de conteneur et une version `1.28.1-2` ou une version ultérieure du package ecs-init, prennent en charge les limites accrues. ENI Si vous utilisez la dernière variante Linux des AMI optimisées pour Amazon ECS, ces exigences seront respectées. Les conteneurs Windows ne sont pas pris en charge à l'heure actuelle.
- Seules les nouvelles EC2 instances Amazon lancées après l'activation `awsvpcTrunking` reçoivent les ENI limites augmentées et l'interface réseau de liaison. Les instances lancées avant cette adoption ne reçoivent pas ces fonctionnalités, quelles que soient les actions réalisées.
- Les requêtes IPv4 DNS basées sur les ressources doivent être désactivées sur les EC2 instances Amazon. Pour désactiver cette option, désactivez l'option Activer les requêtes DNS basées sur les ressources IPV4 (enregistrement A) lorsque vous créez une nouvelle instance dans la console Amazon EC2. Pour désactiver cette option à l'aide de AWS CLI, utilisez la commande suivante.

```
aws ec2 modify-private-dns-name-options --instance-id i-xxxxxxx --no-enable-resource-
name-dns-a-record --no-dry-run
```

- EC2 Les instances Amazon situées dans des sous-réseaux partagés ne sont pas prises en charge. Elles ne pourront pas s'enregistrer dans un cluster si elles sont utilisées.
- Vos tâches doivent utiliser le mode `awsvpc` réseau et le type de EC2 lancement. Les tâches utilisant le type de lancement Fargate ont toujours reçu un message ENI dédié, quel que soit le nombre de tâches lancées. Cette fonctionnalité n'est donc pas nécessaire.
- Vos tâches doivent être lancées dans le même Amazon VPC que votre instance de conteneur. Vos tâches ne parviendront pas à démarrer avec une erreur d'attribut si elles ne sont pas dans le même VPC.
- Lors du lancement d'une nouvelle instance de conteneur, celle-ci passe à l'état `REGISTERING` tandis que l'Elastic Network Interface (ENI) « tronc » est mise en service pour l'instance. Si l'enregistrement échoue, l'instance passe à l'état `REGISTRATION_FAILED`. Vous pouvez dépanner un échec d'enregistrement en décrivant l'instance de conteneur pour afficher le champ `statusReason` qui décrit la raison de l'échec. L'instance de conteneur peut alors être désenregistrée ou résiliée manuellement. Une fois que l'instance de conteneur est désenregistrée ou résiliée avec succès, Amazon ECS supprime le tronc. ENI

> **Note**
>
> Amazon ECS émet des événements de changement d'état d'instance de conteneur que vous pouvez contrôler pour les instances qui passent à un état `REGISTRATION_FAILED`. Pour de plus amples informations, veuillez consulter Événements de modification de l'état de l'instance de conteneur Amazon ECS.

- Une fois que l'instance de conteneur est résiliée, elle passe à l'état `DEREGISTERING` tandis que la mise en service de l'ENI « tronc » est annulée. L'instance passe alors à l'état `INACTIVE`.
- Si une instance de conteneur dans un sous-réseau public avec les ENI limites augmentées est arrêtée puis redémarrée, l'instance perd son adresse IP publique et l'agent de conteneur perd sa connexion.
- Lorsque vous l'activez`awsvpcTrunking`, les instances de conteneur reçoivent un élément supplémentaire ENI qui utilise le groupe de sécurité par défaut du VPC et qui est géré par Amazon ECS.

  Un VPC par défaut est fourni avec un sous-réseau public dans chaque zone de disponibilité, une passerelle Internet et des paramètres permettant la résolution DNS. Le sous-réseau est un sous-réseau public, car la table de routage principale envoie le trafic du sous-réseau destiné à Internet à la passerelle Internet. Pour transformer un sous-réseau par défaut en sous-réseau privé, vous devez supprimer la route 0.0.0.0/0 pointant vers la passerelle Internet. Toutefois, dans ce cas, aucune instance de conteneur exécutée dans ce sous-réseau ne pourra accéder à Internet. Vous pouvez ajouter ou supprimer des règles de groupe de sécurité pour contrôler le trafic entrant et sortant de vos sous-réseaux. Pour plus d'informations, consultez les règles relatives aux groupes de sécurité dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.

Prérequis

Avant de lancer une instance de conteneur avec les ENI limites augmentées, les conditions préalables suivantes doivent être remplies.

- Le rôle lié à un service pour Amazon ECS doit être créé. Le rôle lié à un service Amazon ECS fournit à Amazon ECS les autorisations nécessaires pour passer des appels vers d'autres AWS services en votre nom. Ce rôle est généré automatiquement lorsque vous créez un cluster, ou si vous créez ou mettez à jour un service dans la AWS Management Console. Pour de plus amples

informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS. Vous pouvez également créer le rôle lié à un service à l'aide de la commande suivante AWS CLI .

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- Votre compte ou rôle IAM d'instance de conteneur doit activer le paramètre de compte `awsvpcTrunking`. Nous vous recommandons de créer deux rôles d'instance de conteneur (`ecsInstanceRole`). Vous pouvez ensuite activer le paramètre du `awsvpcTrunking` compte pour un rôle et utiliser ce rôle pour les tâches nécessitant une jonction ENI. Pour plus d'informations sur le rôle d'instance de conteneur, consultezRôle IAM d'instance de conteneur Amazon ECS.

Une fois les conditions requises remplies, vous pouvez lancer une nouvelle instance de conteneur à l'aide de l'un des types d' EC2 instances Amazon pris en charge, et les ENI limites de l'instance seront augmentées. Pour obtenir la liste des types d'instances, consultez Instances prises en charge pour augmenter le nombre d'interfaces réseau de conteneurs Amazon ECS. L'instance de conteneur doit disposer de la version `1.28.1` ou ultérieure de l'agent de conteneur et de la version `1.28.1-2` ou ultérieure du package ecs-init. Si vous utilisez la dernière variante Linux des AMI optimisées pour Amazon ECS, ces exigences seront respectées. Pour de plus amples informations, veuillez consulter Lancement d'une instance de conteneur Amazon ECS Linux.

> **Important**
>
> Les requêtes IPv4 DNS basées sur les ressources doivent être désactivées sur les EC2 instances Amazon. Pour désactiver cette option, assurez-vous que l'option Activer les requêtes DNS basées sur les ressources IPV4 (enregistrement A) est désélectionnée lors de la création d'une nouvelle instance à l'aide de la console Amazon. EC2 Pour désactiver cette option à l'aide de AWS CLI, utilisez la commande suivante.
>
> ```
> aws ec2 modify-private-dns-name-options --instance-id i-xxxxxxx --no-enable-
> resource-name-dns-a-record --no-dry-run
> ```

Pour afficher vos instances de conteneur avec des ENI limites accrues à l'aide du AWS CLI

Chaque instance de conteneur possède une interface réseau par défaut, appelée interface réseau « tronc ». Utilisez la commande suivante pour répertorier vos instances de conteneur dont ENI les

limites sont augmentées en demandant l'`ecs.awsvpc-trunk-id`attribut, qui indique qu'il possède une interface réseau de jonction.

- list-attributes (AWS CLI)

```
aws ecs list-attributes \
      --target-type container-instance \
      --attribute-name ecs.awsvpc-trunk-id \
      --cluster cluster_name \
      --region us-east-1
```

- Obtenir la ECSAttribute liste (AWS Tools for Windows PowerShell)

```
Get-ECSAttributeList -TargetType container-instance -AttributeName ecs.awsvpc-trunk-
id -Region us-east-1
```

Instances prises en charge pour augmenter le nombre d'interfaces réseau de conteneurs Amazon ECS

Vous trouverez ci-dessous les types d' EC2 instances Amazon pris en charge et le nombre de tâches utilisant le mode `awsvpc` réseau qui peuvent être lancées sur chaque type d'instance avant et après l'activation du paramètre du `awsvpcTrunking` compte.

> **Important**
>
> Bien que d'autres types d'instance soient pris en charge dans la même famille d'instances, les types d'instance `a1.metal`, `c5.metal`, `c5a.8xlarge`, `c5ad.8xlarge`, `c5d.metal`, `m5.metal`, `p3dn.24xlarge`, `r5.metal`, `r5.8xlarge` et `r5d.metal` ne sont pas pris en charge.
> Les familles d'instances `c5n`, `d3`, `d3en`, `g3`, `g3s` ,`g4dn`, `i3`, `i3en`, `inf1`, `m5dn`, `m5n`, `m5zn`, `mac1`, `r5b`, `r5n`, `r5dn`, `u-12tb1`, `u-6tb1`, `u-9tb1` et `z1d` ne sont pas prises en charge.

Rubriques

- Usage général
- Calcul optimisé
- Optimisé pour la mémoire
- Stockage optimisé

- Calcul accéléré
- Calcul haute performance

Usage général

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
a1.medium	1	10
a1.large	2	10
a1.xlarge	3	20
a1.2xlarge	3	40
a1.4xlarge	7	60
m5.large	2	10
m5.xlarge	3	20
m5.2xlarge	3	40
m5.4xlarge	7	60
m5.8xlarge	7	60
m5.12xlarge	7	60
m5.16xlarge	14	120
m5.24xlarge	14	120
m5a.large	2	10
m5a.xlarge	3	20
m5a.2xlarge	3	40
m5a.4xlarge	7	60

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
m5a.8xlarge	7	60
m5a.12xlarge	7	60
m5a.16xlarge	14	120
m5a.24xlarge	14	120
m5ad.large	2	10
m5ad.xlarge	3	20
m5ad.2xlarge	3	40
m5ad.4xlarge	7	60
m5ad.8xlarge	7	60
m5ad.12xlarge	7	60
m5ad.16xlarge	14	120
m5ad.24xlarge	14	120
m5d.large	2	10
m5d.xlarge	3	20
m5d.2xlarge	3	40
m5d.4xlarge	7	60
m5d.8xlarge	7	60
m5d.12xlarge	7	60
m5d.16xlarge	14	120
m5d.24xlarge	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
m5d.metal	14	120
m6a.large	2	10
m6a.xlarge	3	20
m6a.2xlarge	3	40
m6a.4xlarge	7	60
m6a.8xlarge	7	90
m6a.12xlarge	7	120
m6a.16xlarge	14	120
m6a.24xlarge	14	120
m6a.32xlarge	14	120
m6a.48xlarge	14	120
m6a.metal	14	120
m6g.medium	1	4
m6g.large	2	10
m6g.xlarge	3	20
m6g.2xlarge	3	40
m6g.4xlarge	7	60
m6g.8xlarge	7	60
m6g.12xlarge	7	60
m6g.16xlarge	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
m6g.metal	14	120
m6gd.medium	1	4
m6gd.large	2	10
m6gd.xlarge	3	20
m6gd.2xlarge	3	40
m6gd.4xlarge	7	60
m6gd.8xlarge	7	60
m6gd.12xlarge	7	60
m6gd.16xlarge	14	120
m6gd.metal	14	120
m6i.large	2	10
m6i.xlarge	3	20
m6i.2xlarge	3	40
m6i.4xlarge	7	60
m6i.8xlarge	7	90
m6i.12xlarge	7	120
m6i.16xlarge	14	120
m6i.24xlarge	14	120
m6i.32xlarge	14	120
m6i.metal	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
m6id.large	2	10
m6id.xlarge	3	20
m6id.2xlarge	3	40
m6id.4xlarge	7	60
m6id.8xlarge	7	90
m6id.12xlarge	7	120
m6id.16xlarge	14	120
m6id.24xlarge	14	120
m6id.32xlarge	14	120
m6id.metal	14	120
m6idn.large	2	10
m6idn.xlarge	3	20
m6idn.2xlarge	3	40
m6idn.4xlarge	7	60
m6idn.8xlarge	7	90
m6idn.12xlarge	7	120
m6idn.16xlarge	14	120
m6idn.24xlarge	14	120
m6idn.32xlarge	15	120
m6idn.metal	15	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
m6in.large	2	10
m6in.xlarge	3	20
m6in.2xlarge	3	40
m6in.4xlarge	7	60
m6in.8xlarge	7	90
m6in.12xlarge	7	120
m6in.16xlarge	14	120
m6in.24xlarge	14	120
m6in.32xlarge	15	120
m6in.metal	15	120
m7a.medium	1	4
m7a.large	2	10
m7a.xlarge	3	20
m7a.2xlarge	3	40
m7a.4xlarge	7	60
m7a.8xlarge	7	90
m7a.12xlarge	7	120
m7a.16xlarge	14	120
m7a.24xlarge	14	120
m7a.32xlarge	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
m7a.48xlarge	14	120
m7a.metal-48xl	14	120
m7g.medium	1	4
m7g.large	2	10
m7g.xlarge	3	20
m7g.2xlarge	3	40
m7g.4xlarge	7	60
m7g.8xlarge	7	60
m7g.12xlarge	7	60
m7g.16xlarge	14	120
m7g.metal	14	120
m7gd.medium	1	4
m7gd.large	2	10
m7gd.xlarge	3	20
m7gd.2xlarge	3	40
m7gd.4xlarge	7	60
m7gd.8xlarge	7	60
m7gd.12xlarge	7	60
m7gd.16xlarge	14	120
m7gd.metal	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
m7i.large	2	10
m7i.xlarge	3	20
m7i.2xlarge	3	40
m7i.4xlarge	7	60
m7i.8xlarge	7	90
m7i.12xlarge	7	120
m7i.16xlarge	14	120
m7i.24xlarge	14	120
m7i.48xlarge	14	120
m7i.metal-24xl	14	120
m7i.metal-48xl	14	120
m7i-flex.large	2	4
m7i-flex.xlarge	3	10
m7i-flex.2xlarge	3	20
m7i-flex.4xlarge	7	40
m7i-flex.8xlarge	7	60
M7i-Flex. 12 x large	7	120
m7i-flex. 16 x large	14	120
8 mg. moyen	1	4
8 mg de large	2	10

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
8 mg.xlarge	3	20
8 kg x 2 x large	3	40
8 kg x 4 x large	7	60
8 g x 8 x large	7	60
8 kg x 12 x large	7	60
8 g x 16 x large	14	120
8 g, 24 x large	14	120
8 g x 48 x large	14	120
8 mg.metal-24xl	14	120
8 mg.metal-48xl	14	120
m8gd.medium	1	4
8 mgd.large	2	10
m8gd.xlarge	3	20
8 mm. 2 x large	3	40
8 mm x 4 x large	7	60
8 mm x 8 x large	7	60
8 mm x 12 x large	7	60
8 mm x 16 x large	14	120
8 mm x 24 x large	14	120
8 mm x 48 x large	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
m8gd.metal-24xl	14	120
m8gd.metal-48xl	14	120
mac2.metal	7	12
mac2-m1 ultra metal	7	12
mac2-m2.metal	7	12
mac2-m2pro.metal	7	12

Calcul optimisé

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c5.large	2	10
c5.xlarge	3	20
c5.2xlarge	3	40
c5.4xlarge	7	60
c5.9xlarge	7	60
c5.12xlarge	7	60
c5.18xlarge	14	120
c5.24xlarge	14	120
c5a.large	2	10
c5a.xlarge	3	20

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c5a.2xlarge	3	40
c5a.4xlarge	7	60
c5a.12xlarge	7	60
c5a.16xlarge	14	120
c5a.24xlarge	14	120
c5ad.large	2	10
c5ad.xlarge	3	20
c5ad.2xlarge	3	40
c5ad.4xlarge	7	60
c5ad.12xlarge	7	60
c5ad.16xlarge	14	120
c5ad.24xlarge	14	120
c5d.large	2	10
c5d.xlarge	3	20
c5d.2xlarge	3	40
c5d.4xlarge	7	60
c5d.9xlarge	7	60
c5d.12xlarge	7	60
c5d.18xlarge	14	120
c5d.24xlarge	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c6a.large	2	10
c6a.xlarge	3	20
c6a.2xlarge	3	40
c6a.4xlarge	7	60
c6a.8xlarge	7	90
c6a.12xlarge	7	120
c6a.16xlarge	14	120
c6a.24xlarge	14	120
c6a.32xlarge	14	120
c6a.48xlarge	14	120
c6a.metal	14	120
c6g.medium	1	4
c6g.large	2	10
c6g.xlarge	3	20
c6g.2xlarge	3	40
c6g.4xlarge	7	60
c6g.8xlarge	7	60
c6g.12xlarge	7	60
c6g.16xlarge	14	120
c6g.metal	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c6gd.medium	1	4
c6gd.large	2	10
c6gd.xlarge	3	20
c6gd.2xlarge	3	40
c6gd.4xlarge	7	60
c6gd.8xlarge	7	60
c6gd.12xlarge	7	60
c6gd.16xlarge	14	120
c6gd.metal	14	120
c6gn.medium	1	4
c6gn.large	2	10
c6gn.xlarge	3	20
c6gn.2xlarge	3	40
c6gn.4xlarge	7	60
c6gn.8xlarge	7	60
c6gn.12xlarge	7	60
c6gn.16xlarge	14	120
c6i.large	2	10
c6i.xlarge	3	20
c6i.2xlarge	3	40

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c6i.4xlarge	7	60
c6i.8xlarge	7	90
c6i.12xlarge	7	120
c6i.16xlarge	14	120
c6i.24xlarge	14	120
c6i.32xlarge	14	120
c6i.metal	14	120
c6id.large	2	10
c6id.xlarge	3	20
c6id.2xlarge	3	40
c6id.4xlarge	7	60
c6id.8xlarge	7	90
c6id.12xlarge	7	120
c6id.16xlarge	14	120
c6id.24xlarge	14	120
c6id.32xlarge	14	120
c6id.metal	14	120
c6in.large	2	10
c6in.xlarge	3	20
c6in.2xlarge	3	40

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c6in.4xlarge	7	60
c6in.8xlarge	7	90
c6in.12xlarge	7	120
c6in.16xlarge	14	120
c6in.24xlarge	14	120
c6in.32xlarge	15	120
c6in.metal	15	120
c7a.medium	1	4
c7a.large	2	10
c7a.xlarge	3	20
c7a.2xlarge	3	40
c7a.4xlarge	7	60
c7a.8xlarge	7	90
c7a.12xlarge	7	120
c7a.16xlarge	14	120
c7a.24xlarge	14	120
c7a.32xlarge	14	120
c7a.48xlarge	14	120
c7a.metal-48xl	14	120
c7g.medium	1	4

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c7g.large	2	10
c7g.xlarge	3	20
c7g.2xlarge	3	40
c7g.4xlarge	7	60
c7g.8xlarge	7	60
c7g.12xlarge	7	60
c7g.16xlarge	14	120
c7g.metal	14	120
c7gd.medium	1	4
c7gd.large	2	10
c7gd.xlarge	3	20
c7gd.2xlarge	3	40
c7gd.4xlarge	7	60
c7gd.8xlarge	7	60
c7gd.12xlarge	7	60
c7gd.16xlarge	14	120
c7gd.metal	14	120
c7gn.medium	1	4
c7gn.large	2	10
c7gn.xlarge	3	20

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c7gn.2xlarge	3	40
c7gn.4xlarge	7	60
c7gn.8xlarge	7	60
c7gn.12xlarge	7	60
c7gn.16xlarge	14	120
c7gn.metal	14	120
c7i.large	2	10
c7i.xlarge	3	20
c7i.2xlarge	3	40
c7i.4xlarge	7	60
c7i.8xlarge	7	90
c7i.12xlarge	7	120
c7i.16xlarge	14	120
c7i.24xlarge	14	120
c7i.48xlarge	14	120
c7i.metal-24xl	14	120
c7i.metal-48xl	14	120
c7i-flex.large	2	4
c7i-flex.xlarge	3	10
c7i-flex. 2 x large	3	20

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c7i-flex 4 x large	7	40
c7i-flex 8 x large	7	60
c7i-flex. 12 x large	7	120
c7i-flex. 16 x large	14	120
c8g .medium	1	4
8 g de large	2	10
c8g.xlarge	3	20
8 cm x 2 x large	3	40
8 cm x 4 x large	7	60
8 cm x 8 x large	7	60
8 g, 12 x large	7	60
8 g x 16 x large	14	120
8 g, 24 x large	14	120
8 g x 48 x large	14	120
c8g.metal-24xl	14	120
c8g.metal-48xl	14	120
c8gd.medium	1	4
c8gd.large	2	10
c8gd.xlarge	3	20
c8gd.2xlarge	3	40

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c8gd.4xlarge	7	60
8 cm x 8 x large	7	60
8 cm x 12 x large	7	60
8 cm x 16 x large	14	120
c8g. 24 x large	14	120
c8g x 48 x large	14	120
c8gd.metal-24xl	14	120
c8gd.metal-48xl	14	120
c8gn.medium	1	4
c8gn.large	2	10
c8gn.xlarge	3	20
8 cm, 2 x large	3	40
8 cm x 4 x large	7	60
8 cm x 8 x large	9	60
8 cm x 12 x large	11	60
8 cm x 16 x large	15	120
c8g. 24 x large	23	120
8 cm x 48 x large	23	120
c8gn.metal-24xl	23	120
c8gn.metal-48xl	23	120

Optimisé pour la mémoire

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
r5.large	2	10
r5.xlarge	3	20
r5.2xlarge	3	40
r5.4xlarge	7	60
r5.12xlarge	7	60
r5.16xlarge	14	120
r5.24xlarge	14	120
r5a.large	2	10
r5a.xlarge	3	20
r5a.2xlarge	3	40
r5a.4xlarge	7	60
r5a.8xlarge	7	60
r5a.12xlarge	7	60
r5a.16xlarge	14	120
r5a.24xlarge	14	120
r5ad.large	2	10
r5ad.xlarge	3	20
r5ad.2xlarge	3	40
r5ad.4xlarge	7	60

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
r5ad.8xlarge	7	60
r5ad.12xlarge	7	60
r5ad.16xlarge	14	120
r5ad.24xlarge	14	120
r5b.16xlarge	14	120
r5d.large	2	10
r5d.xlarge	3	20
r5d.2xlarge	3	40
r5d.4xlarge	7	60
r5d.8xlarge	7	60
r5d.12xlarge	7	60
r5d.16xlarge	14	120
r5d.24xlarge	14	120
r5dn.16xlarge	14	120
r6a.large	2	10
r6a.xlarge	3	20
r6a.2xlarge	3	40
r6a.4xlarge	7	60
r6a.8xlarge	7	90
r6a.12xlarge	7	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
r6a.16xlarge	14	120
r6a.24xlarge	14	120
r6a.32xlarge	14	120
r6a.48xlarge	14	120
r6a.metal	14	120
r6g.medium	1	4
r6g.large	2	10
r6g.xlarge	3	20
r6g.2xlarge	3	40
r6g.4xlarge	7	60
r6g.8xlarge	7	60
r6g.12xlarge	7	60
r6g.16xlarge	14	120
r6g.metal	14	120
r6gd.medium	1	4
r6gd.large	2	10
r6gd.xlarge	3	20
r6gd.2xlarge	3	40
r6gd.4xlarge	7	60
r6gd.8xlarge	7	60

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
r6gd.12xlarge	7	60
r6gd.16xlarge	14	120
r6gd.metal	14	120
r6i.large	2	10
r6i.xlarge	3	20
r6i.2xlarge	3	40
r6i.4xlarge	7	60
r6i.8xlarge	7	90
r6i.12xlarge	7	120
r6i.16xlarge	14	120
r6i.24xlarge	14	120
r6i.32xlarge	14	120
r6i.metal	14	120
r6idn.large	2	10
r6idn.xlarge	3	20
r6idn.2xlarge	3	40
r6idn.4xlarge	7	60
r6idn.8xlarge	7	90
r6idn.12xlarge	7	120
r6idn.16xlarge	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
r6idn.24xlarge	14	120
r6idn.32xlarge	15	120
r6idn.metal	15	120
r6in.large	2	10
r6in.xlarge	3	20
r6in.2xlarge	3	40
r6in.4xlarge	7	60
r6in.8xlarge	7	90
r6in.12xlarge	7	120
r6in.16xlarge	14	120
r6in.24xlarge	14	120
r6in.32xlarge	15	120
r6in.metal	15	120
r6id.large	2	10
r6id.xlarge	3	20
r6id.2xlarge	3	40
r6id.4xlarge	7	60
r6id.8xlarge	7	90
r6id.12xlarge	7	120
r6id.16xlarge	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
r6id.24xlarge	14	120
r6id.32xlarge	14	120
r6id.metal	14	120
r7a.medium	1	4
r7a.large	2	10
r7a.xlarge	3	20
r7a.2xlarge	3	40
r7a.4xlarge	7	60
r7a.8xlarge	7	90
r7a.12xlarge	7	120
r7a.16xlarge	14	120
r7a.24xlarge	14	120
r7a.32xlarge	14	120
r7a.48xlarge	14	120
r7a.metal-48xl	14	120
r7g.medium	1	4
r7g.large	2	10
r7g.xlarge	3	20
r7g.2xlarge	3	40
r7g.4xlarge	7	60

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
r7g.8xlarge	7	60
r7g.12xlarge	7	60
r7g.16xlarge	14	120
r7g.metal	14	120
r7gd.medium	1	4
r7gd.large	2	10
r7gd.xlarge	3	20
r7gd.2xlarge	3	40
r7gd.4xlarge	7	60
r7gd.8xlarge	7	60
r7gd.12xlarge	7	60
r7gd.16xlarge	14	120
r7gd.metal	14	120
r7i.large	2	10
r7i.xlarge	3	20
r7i.2xlarge	3	40
r7i.4xlarge	7	60
r7i.8xlarge	7	90
r7i.12xlarge	7	120
r7i.16xlarge	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
r7i.24xlarge	14	120
r7i.48xlarge	14	120
r7i.metal-24xl	14	120
r7i.metal-48xl	14	120
r7iz.large	2	10
r7iz.xlarge	3	20
r7iz.2xlarge	3	40
r7iz.4xlarge	7	60
r7iz.8xlarge	7	90
r7iz.12xlarge	7	120
r7iz.16xlarge	14	120
r7iz.32xlarge	14	120
r7iz.metal-16xl	14	120
r7iz.metal-32xl	14	120
r8g. medium	1	4
8 g. large	2	10
r8g.xlarge	3	20
8 g x 2 x large	3	40
8 g x 4 x large	7	60
8 g x 8 x large	7	60

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
8 g, 12 x large	7	60
8 g, 16 x large	14	120
8 g, 24 x large	14	120
8 g, 48 x large	14	120
r8g.metal-24xl	14	120
r8g.metal-48xl	14	120
r8gd.medium	1	4
r8gd.large	2	10
r8gd.xlarge	3	20
r8gd.2xlarge	3	40
r8gd.4xlarge	7	60
8 g x 8 x large	7	60
8 g x 12 x large	7	60
8 g x 16 x large	14	120
8 g x 24 x large	14	120
8 g x 48 x large	14	120
r8gd.metal-24xl	14	120
r8gd.metal-48xl	14	120
u-3tb1.56xlarge	7	12
u-6tb1.56xlarge	14	12

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
u-18tb1.112xlarge	14	12
u-18tb1.metal	14	12
u-24tb1.112xlarge	14	12
u-24tb1.metal	14	12
u7i-6tb.112 x large	14	120
u7i-8tb.112 x large	14	120
u7i-12tb.224 x large	14	120
U7 en 16 TB, 224 x large	15	120
U7 en 24 TB, 224 x large	15	120
U7 en 32 TB, 224 x large	15	120
U7 pouces - 32 TB, 480 x large	15	120
x2gd.medium	1	10
x2gd.large	2	10
x2gd.xlarge	3	20
x2gd.2xlarge	3	40
x2gd.4xlarge	7	60
x2gd.8xlarge	7	60
x2gd.12xlarge	7	60
x2gd.16xlarge	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
x2gd.metal	14	120
x2idn.16xlarge	14	120
x2idn.24xlarge	14	120
x2idn.32xlarge	14	120
x2idn.metal	14	120
x2iedn.xlarge	3	13
x2iedn.2xlarge	3	29
x2iedn.4xlarge	7	60
x2iedn.8xlarge	7	120
x2iedn.16xlarge	14	120
x2iedn.24xlarge	14	120
x2iedn.32xlarge	14	120
x2iedn.metal	14	120
x2iezn.2xlarge	3	64
x2iezn.4xlarge	7	120
x2iezn.6xlarge	7	120
x2iezn.8xlarge	7	120
x2iezn.12xlarge	14	120
x2iezn.metal	14	120
8 x 8 g, taille moyenne	1	4

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
8 g de large	2	10
x 8 g x large	3	20
8 x g, 2 x large	3	40
8 x 8 x 4 x large	7	60
8 x 8 x 8 x large	7	60
8 x 12 x large	7	60
8 x 16 x large	14	120
8 g x 24 x large	14	120
8 g x 48 x large	14	120
8 g, métal, 24 XL	14	120
8 g, métal, 48 xl	14	120

Stockage optimisé

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
i4g.large	2	10
i4g.xlarge	3	20
i4g.2xlarge	3	40
i4g.4xlarge	7	60
i4g.8xlarge	7	60

Type d’instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
i4g.16xlarge	14	120
i4i.xlarge	3	8
i4i.2xlarge	3	28
i4i.4xlarge	7	58
i4i.8xlarge	7	118
i4i.12xlarge	7	118
i4i.16xlarge	14	248
i4i.24xlarge	14	118
i4i.32xlarge	14	498
i4i.metal	14	498
i7i.large	2	10
i7i.xlarge	3	20
i7i.2xlarge	3	40
i7i.4xlarge	7	60
i7 x 8 x large	7	90
i7i 12 x large	7	90
i7i 16 x large	14	120
i7i 24 x large	14	120
i7i 48 x large	14	120
i7i.metal-24xl	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
i7i.metal-48xl	14	120
i7ie.large	2	20
i7ie.xlarge	3	29
i7ie.2xlarge	3	29
i7ie. 3 x large	3	29
i7ie. 6 x large	7	60
i7ie. 12 x large	7	60
i7ie. 18 x large	14	120
i7ie. 24 x large	14	120
i7ie. 48 x large	14	120
i7ie.metal-24xl	14	120
i7ie.metal-48xl	14	120
8 g de large	2	10
i8g.xlarge	3	20
8 g, 2 x large	3	40
8 g x 4 x large	7	60
8 g x 8 x large	7	60
8 g, 12 x large	7	60
8 g, 16 x large	14	120
8 g, 24 x large	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
8 g, 48 x large	14	120
i8g.metal-24xl	14	120
im4gn.large	2	10
im4gn.xlarge	3	20
im4gn.2xlarge	3	40
im4gn.4xlarge	7	60
im4gn.8xlarge	7	60
im4gn.16xlarge	14	120
is4gen.medium	1	4
is4gen.large	2	10
is4gen.xlarge	3	20
is4gen.2xlarge	3	40
is4gen.4xlarge	7	60
is4gen.8xlarge	7	60

Calcul accéléré

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
dl1.24xlarge	59	120
dl2q.24xlarge	14	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
F2,6 x large	7	90
f/2,12 x large	7	120
f 2,48 x large	14	120
g4ad.xlarge	1	12
g4ad.2xlarge	1	12
g4ad.4xlarge	2	12
g4ad.8xlarge	3	12
g4ad.16xlarge	7	12
g5.xlarge	3	6
g5.2xlarge	3	19
g5.4xlarge	7	40
g5.8xlarge	7	90
g5.12xlarge	14	120
g5.16xlarge	7	120
g5.24xlarge	14	120
g5.48xlarge	6	120
g5g.xlarge	3	20
g5g.2xlarge	3	40
g5g.4xlarge	7	60
g5g.8xlarge	7	60

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
g5g.16xlarge	14	120
g5g.metal	14	120
g6.xlarge	3	20
g 6,2 x large	3	40
g 6,4 x large	7	60
g 6,8 x large	7	90
g 6,12 x large	7	120
g 6,16 x large	14	120
g 6,24 x large	14	120
g 6,48 x large	14	120
g6e.xlarge	3	20
g6e.2xlarge	3	40
g6e.4xlarge	7	60
6 x 8 x large	7	90
G6E, 12 x large	9	120
G6E, 16 x large	14	120
G6E, 24 x large	19	120
g 6 e 48 x large	39	120
G6f. Large	1	10
g6f.xlarge	3	20

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
g6f.2xlarge	3	40
6 f x 4 x large	7	60
gr6,4 x large	7	60
gr6,8 x large	7	90
gr6f. 4 x large	7	60
inf2.xlarge	3	20
inf2.8xlarge	7	90
inf2.24xlarge	14	120
inf2.48xlarge	14	120
p4d.24xlarge	59	120
p4de.24xlarge	59	120
p5.48xlarge	63	242
p5e.48 x large	63	242
p 5 en 48 x large	63	242
p6-b200.48 x large	31	242
p6e-gb 200,36 x large	38	120
trn1.2xlarge	3	19
trn1.32xlarge	39	120
trn1n.32xlarge	79	242
trn 2,48 x large	31	242

Type d’instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
trn2u 48 x large	31	242
vt1.3xlarge	3	40
vt1.6xlarge	7	60
vt1.24xlarge	14	120

Calcul haute performance

Type d’instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
hpc6a.48xlarge	1	120
hpc6id.32xlarge	1	120
hpc7g.4xlarge	3	120
hpc7g.8xlarge	3	120
hpc7g.16xlarge	3	120

Réservation de mémoire d'instance de conteneur Amazon ECS Linux

Lorsque l'agent de conteneur Amazon ECS enregistre une instance de conteneur dans un cluster, il doit déterminer la quantité de mémoire disponible que l'instance de conteneur peut réserver pour vos tâches. En raison de la surcharge de mémoire de la plate-forme et de la mémoire occupée par le noyau du système, ce nombre est différent de la quantité de mémoire installée annoncée pour les EC2 instances Amazon. Par exemple, une instance `m4.large` a 8 Gio de mémoire installée. Cependant, cela ne se traduit pas toujours par exactement 8192 MiB de mémoire disponible pour les tâches lors de l'enregistrement de l'instance de conteneur.

L'agent de conteneur Amazon ECS fournit une variable de configuration appelée`ECS_RESERVED_MEMORY`, que vous pouvez utiliser pour supprimer un nombre spécifié de

MiB de mémoire du pool alloué à vos tâches. Cela réserve de manière effective cette quantité de mémoire pour les processus système critiques.

Si vos tâches occupent toute la mémoire d'une instance de conteneur, il est possible que celles-ci soient confrontées à des processus critiques du système en termes de mémoire et qu'elles soient à l'origine d'une défaillance du système.

Par exemple, si vous spécifiez `ECS_RESERVED_MEMORY=256` dans votre fichier de configuration de l'agent de conteneur, l'agent enregistre la mémoire totale, moins 256 Mio pour cette instance. 256 Mio de mémoire ne sont donc pas allouées par des tâches ECS. Pour plus d'informations sur les variables de configuration de l'agent et la façon de les définir, consultez Configuration de l'agent de conteneur Amazon ECS et Démarrage des instances de conteneur Linux Amazon ECS pour transmettre des données.

Si vous spécifiez 8192 Mo pour la tâche et qu'aucune de vos instances de conteneur ne dispose de 8192 Mo de mémoire disponible ou plus pour satisfaire cette exigence, la tâche ne peut pas être placée dans votre cluster. Si vous utilisez un environnement informatique géré, vous AWS Batch devez lancer un type d'instance plus important pour répondre à la demande.

Vous devez également réserver une partie de la mémoire pour l'agent de conteneur Amazon ECS et d'autres processus système critiques sur vos instances de conteneur, afin que les conteneurs de votre tâche ne se disputent pas la même mémoire, ce qui pourrait provoque une défaillance du système.

L'agent de conteneur Amazon ECS utilise la fonction `ReadMemInfo()` Docker pour connaître la mémoire totale disponible pour le système d'exploitation. Linux et Windows fournissent tous deux des utilitaires de ligne de commande pour déterminer la mémoire totale.

Example - Déterminer la mémoire totale Linux

La commande free renvoie la mémoire totale reconnue par le système d'exploitation.

```
$ free -b
```

Exemple de sortie d'une instance `m4.large` exécutant l'AMI Amazon Linux optimisée pour Amazon ECS.

```
             total       used       free     shared    buffers     cached
Mem:    8373026816  348180480 8024846336      90112   25534464  205418496
-/+ buffers/cache:  117227520 8255799296
```

Cette instance comporte 8373026816 octets de mémoire totale, soit 7 985 Mio disponible pour les tâches.

Example - Déterminer la mémoire totale Windows

La commande wmic renvoie la mémoire totale reconnue par le système d'exploitation.

```
C:\> wmic ComputerSystem get TotalPhysicalMemory
```

Exemple de sortie pour une `m4.large` instance exécutant l'AMI Windows Server optimisée pour Amazon ECS.

```
TotalPhysicalMemory
8589524992
```

Cette instance comporte 8589524992 octets de mémoire totale, soit 8 191 Mio disponible pour les tâches.

Affichage de la mémoire d'une instance de conteneur

Vous pouvez voir la quantité de mémoire utilisée par une instance de conteneur dans la console Amazon ECS (ou dans le cadre de l'opération DescribeContainerInstancesAPI). Si vous essayez de maximiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, vous pouvez observer la mémoire disponible pour cette instance de conteneur, puis attribuer cette quantité de mémoire à vos tâches.

Pour afficher la mémoire d'une instance de conteneur

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le volet de navigation, choisissez Clusters, puis choisissez le cluster qui héberge votre instance de conteneur.
3. Choisissez Infrastructure, puis sous Instances de conteneur, choisissez une instance de conteneur.
4. La section Ressources indique la mémoire enregistrée et disponible pour l'instance de conteneur.

   La valeur de mémoire enregistrée correspond à l'instance de conteneur ; enregistrée auprès d'Amazon ECS lors de son premier lancement, et la valeur de mémoire disponible correspond à ce qui n'a pas encore été alloué aux tâches.

Gestion à distance des instances de conteneur Amazon ECS à l'aide de AWS Systems Manager

Vous pouvez utiliser la fonctionnalité Run Command dans AWS Systems Manager (Systems Manager) pour gérer de manière sécurisée et à distance la configuration de vos instances de conteneur Amazon ECS. Exécutez la commande vous fournit des outils simples afin d'effectuer des tâches administratives courantes sans qu'il soit nécessaire de se connecter localement à l'instance. Vous pouvez gérer les changements de configuration sur vos clusters en exécutant des commandes simultanément sur plusieurs instances de conteneur. La fonctionnalité Exécuter la commande affiche le statut et les résultats de chaque commande.

Voici quelques exemples des types de tâches susceptibles d'être effectués avec la fonctionnalité Exécuter la commande :

- Installer ou désinstaller des packages
- Effectuer des mises à jour de sécurité
- Nettoyer les images Docker
- Arrêter ou démarrer des services
- Afficher les ressources système
- Afficher les fichiers journaux
- Effectuer des opérations sur les fichiers

Pour plus d'informations sur Run Command, veuillez consulter AWS Systems Manager Run Command dans le Guide de l'utilisateur AWS Systems Manager .

Les conditions suivantes sont requises pour utiliser Systems Manager avec Amazon ECS.

1. Vous devez accorder à l'instance de conteneur role (ecsInstanceRole) les autorisations d'accès au Systems Manager APIs. Vous pouvez le faire en affectant l'Amazon SSMManaged InstanceCore au `ecsInstanceRole` rôle. Pour plus d'informations sur la façon d'associer une politique à un rôle, voir Mettre à jour les autorisations pour un rôle dans le Guide de AWS Identity and Access Management l'utilisateur
2. Vérifier que SSM Agent est installé sur vos instances de conteneur. Pour plus d'informations, voir Installation et désinstallation manuelles de l'agent SSM sur les EC2 instances pour Linux.

Après avoir attaché les politiques gérées par Systems Manager à vos instances de conteneur `ecsInstanceRole` et vérifié que AWS Systems Manager l'agent (agent SSM) est installé sur

vos instances de conteneur, vous pouvez commencer à utiliser Run Command pour envoyer des commandes à vos instances de conteneur. Pour de plus amples informations sur l'exécution des commandes et des scripts shell sur vos instances et pour afficher la sortie obtenue, veuillez consulter Exécution de commandes avec Systems Manager Run Command et Procédures Run Command dans le Guide de l'utilisateur AWS Systems Manager .

Un cas d'utilisation courant consiste à mettre à jour le logiciel d'instance de conteneur avec Run Command. Vous pouvez suivre les procédures décrites dans le guide de l' AWS Systems Manager utilisateur avec les paramètres suivants.

Paramètre	Valeur
Document de commande	`AWS-RunShellScript`
Commande	`$ yum update -y`
Instances cibles	Vos instances de conteneur

Utilisation d'un proxy HTTP pour les instances de conteneur Amazon ECS Linux

Vous pouvez configurer vos instances de conteneur Amazon ECS pour qu'elles utilisent un proxy HTTP pour l'agent de conteneur Amazon ECS et le démon Docker. C'est utile si vos instances de conteneur n'ont pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

Pour configurer votre instance de conteneur Amazon ECS Linux afin d'utiliser un proxy HTTP, définissez les variables suivantes dans les fichiers concernés au moment du lancement (avec les données EC2 utilisateur Amazon). Vous pouvez également modifier manuellement le fichier de configuration, puis redémarrer l'agent.

`/etc/ecs/ecs.config`(Amazon Linux 2et AmazonLinux AMI)

```
HTTP_PROXY=10.0.0.131:3128
```

Pour cette valeur, employez le nom d'hôte (ou l'adresse IP) et le numéro de port d'un proxy HTTP à utiliser pour que l'agent Amazon ECS puisse se connecter à Internet. Par exemple, vos instances de conteneur n'ont peut-être pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

`NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock`

Définissez cette valeur sur `169.254.169.254,169.254.170.2,/var/run/docker.sock` pour filtrer les métadonnées de l' EC2 instance, les rôles IAM pour les tâches et le trafic du daemon Docker provenant du proxy.

`/etc/systemd/system/ecs.service.d/http-proxy.conf` (Amazon Linux 2 uniquement)

`Environment="HTTP_PROXY=10.0.0.131:3128/"`

Pour cette valeur, employez le nom d'hôte (ou l'adresse IP) et le numéro de port d'un proxy HTTP à utiliser pour que `ecs-init` puisse se connecter à Internet. Par exemple, vos instances de conteneur n'ont peut-être pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

`Environment="NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock"`

Définissez cette valeur sur `169.254.169.254,169.254.170.2,/var/run/docker.sock` pour filtrer les métadonnées de l' EC2 instance, les rôles IAM pour les tâches et le trafic du daemon Docker provenant du proxy.

`/etc/init/ecs.override` (Amazon Linux AMI uniquement)

`env HTTP_PROXY=10.0.0.131:3128`

Pour cette valeur, employez le nom d'hôte (ou l'adresse IP) et le numéro de port d'un proxy HTTP à utiliser pour que `ecs-init` puisse se connecter à Internet. Par exemple, vos instances de conteneur n'ont peut-être pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

`env NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock`

Définissez cette valeur sur `169.254.169.254,169.254.170.2,/var/run/docker.sock` pour filtrer les métadonnées de l' EC2 instance, les rôles IAM pour les tâches et le trafic du daemon Docker provenant du proxy.

`/etc/systemd/system/docker.service.d/http-proxy.conf` (Amazon Linux 2 uniquement)

`Environment="HTTP_PROXY=http://10.0.0.131:3128"`

Pour cette valeur, employez le nom d'hôte (ou l'adresse IP) et le numéro de port d'un proxy HTTP à utiliser pour que le démon Docker puisse se connecter à Internet. Par exemple, vos instances de conteneur n'ont peut-être pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

`Environment="NO_PROXY=169.254.169.254,169.254.170.2"`

Définissez cette valeur sur `169.254.169.254,169.254.170.2` pour filtrer les métadonnées de l' EC2instance depuis le proxy.

`/etc/sysconfig/docker` (Amazon Linux AMI et Amazon Linux 2 uniquement)

`export HTTP_PROXY=http://10.0.0.131:3128`

Pour cette valeur, employez le nom d'hôte (ou l'adresse IP) et le numéro de port d'un proxy HTTP à utiliser pour que le démon Docker puisse se connecter à Internet. Par exemple, vos instances de conteneur n'ont peut-être pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

`export NO_PROXY=169.254.169.254,169.254.170.2`

Définissez cette valeur sur `169.254.169.254,169.254.170.2` pour filtrer les métadonnées de l' EC2instance depuis le proxy.

La définition de ces variables d'environnement dans les fichiers ci-dessus affecte uniquement l'agent de conteneur Amazon ECS, `ecs-init`, et le démon Docker. Elles ne configurent aucun autre service (comme yum) pour utiliser le proxy.

Pour plus d'informations sur la configuration du proxy, consultez Comment configurer un proxy HTTP pour Docker et l'agent de conteneur Amazon ECS dans Amazon Linux 2 ou AL2 023.

Configuration d'instances pré-initialisées pour votre groupe Amazon ECS Auto Scaling

Amazon ECS prend en charge les pools chauds Amazon EC2 Auto Scaling. Un warm pool est un groupe d' EC2 instances Amazon pré-initialisées prêtes à être mises en service. Chaque fois que votre application doit évoluer, Amazon EC2 Auto Scaling utilise les instances préinitialisées issues du pool de chaleur plutôt que de lancer des instances froides, autorise l'exécution de tout processus d'initialisation final, puis met l'instance en service.

Pour en savoir plus sur les pools chauds et sur la façon d'ajouter un pool chaud à votre groupe Auto Scaling, consultez la section Warm pools for Amazon EC2 Auto Scaling dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling.

Lorsque vous créez ou mettez à jour un groupe instances pré-initialisées pour un groupe Auto Scaling pour Amazon ECS, vous ne pouvez pas définir l'option qui renvoie les instances vers le groupe instances pré-initialisées à mise à l'échelle horizontale (`ReuseOnScaleIn`). Pour plus d'informations, consultez put-warm-pool dans la référence AWS Command Line Interface .

Pour utiliser des pools chauds avec votre cluster Amazon ECS, définissez la variable de configuration de l'`ECS_WARM_POOLS_CHECK`agent sur `true` dans le champ User data de votre modèle de lancement de groupe Amazon EC2 Auto Scaling.

Voici un exemple de la manière dont la variable de configuration de l'agent peut être spécifiée dans le champ Données utilisateur d'un modèle de EC2 lancement Amazon. Remplacez *`MyCluster`* par le nom de votre cluster.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_WARM_POOLS_CHECK=true
EOF
```

La variable `ECS_WARM_POOLS_CHECK` est uniquement prise en charge sur les versions d'agent `1.59.0` et ultérieures. Pour plus d'informations sur la variable, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

Mise à jour de l'agent de conteneur Amazon ECS

Il se peut que vous deviez parfois mettre à jour l'agent de conteneur Amazon ECS pour corriger des bogues et intégrer de nouvelles fonctionnalités. La mise à jour de l'agent de conteneur Amazon ECS n'interrompt pas les tâches ou les services en cours d'exécution sur l'instance de conteneur. Le processus de mise à jour de l'agent varie selon si votre instance de conteneur a été lancée avec l'AMI optimisée pour Amazon ECS ou un autre système d'exploitation.

> **Note**
>
> Les mises à jour de l'agent ne s'appliquent pas aux instances de conteneur Windows. Nous vous recommandons de lancer de nouvelles instances de conteneur pour mettre à jour le version de l'agent dans vos clusters Windows.

Vérification de la version d'agent de conteneur Amazon ECS

Vous pouvez vérifier la version de l'agent de conteneur qui est en cours d'exécution sur vos instances de conteneur afin de savoir s'il doit être mis à jour. L'affichage d'une instance de conteneur dans la console Amazon ECS fournit la version de l'agent. Procédez comme indiqué ci-dessous pour vérifier la version de votre agent.

Amazon ECS console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, choisissez la région dans laquelle votre instance externe est inscrite.
3. Dans le panneau de navigation, choisissez Clusters, puis sélectionnez le cluster qui héberge l'instance externe.
4. Sur la *name* page Cluster :, choisissez l'onglet Infrastructure.
5. Sous Container instances (Instances de conteneur), notez la colonne Agent version (Version d'agent) pour vos instances de conteneur. Si l'instance de conteneur ne contient pas la dernière version de l'agent de conteneur, la console vous prévient à l'aide d'un message et signale la version d'agent obsolète.

   Si votre version d'agent est obsolète, vous pouvez la mettre à jour en suivant les instructions ci-dessous :

   - Si votre instance de conteneur exécute l'AMI optimisée pour Amazon ECS, veuillez consulter Mise à jour de l'agent de conteneur Amazon ECS sur une AMI optimisée pour Amazon ECS.
   - Si votre instance de conteneur n'exécute pas une AMI optimisée pour Amazon ECS, veuillez consulter Mise à jour manuelle de l'agent de conteneur Amazon ECS (pour les applications non optimisées pour Amazon ECS AMIs).

   > **Important**
   >
   > Pour mettre à jour la version de l'agent Amazon ECS à partir des versions antérieures à v1.0.0 sur votre AMI optimisée pour Amazon ECS, nous vous recommandons de résilier votre instance de conteneur actuelle et de lancer une nouvelle instance avec la version d'AMI la plus récente. Toutes les instances de conteneur qui utilisent une version précédente doivent être retirées et remplacées par l'AMI la plus récente. Pour de plus amples informations, veuillez consulter Lancement d'une instance de conteneur Amazon ECS Linux.

Amazon ECS container agent introspection API

Vous pouvez également utiliser l'API d'introspection d'agent de conteneur Amazon ECS pour vérifier la version de l'agent à partir de l'instance de conteneur elle-même. Pour de plus amples informations, veuillez consulter Introspection des conteneurs Amazon ECS.

Pour vérifier si votre agent de conteneur Amazon ECS exécute la dernière version avec l'API d'introspection

1. Connectez-vous à votre instance de conteneur via SSH.
2. Interrogez l'API d'introspection.

```
[ec2-user ~]$ curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
```

> **Note**
>
> L'API d'introspection a ajouté les informations `Version` dans la version v1.0.0 de l'agent de conteneur Amazon ECS. Si `Version` n'apparaît pas lors de l'interrogation de l'API d'introspection ou que l'API d'introspection n'apparaît pas du tout dans votre agent, la version que vous exécutez est v0.0.3 ou une version antérieure. Vous devez mettre à jour votre version.

Mise à jour de l'agent de conteneur Amazon ECS sur une AMI optimisée pour Amazon ECS

Si vous utilisez l'AMI optimisée pour Amazon ECS, vous disposez de plusieurs options pour obtenir la dernière version de l'agent de conteneur Amazon ECS (présentées dans l'ordre de recommandation) :

- Résiliez l'instance de conteneur et lancez la dernière version de l'AMI Amazon Linux 2 optimisée pour Amazon ECS (manuellement ou en mettant à jour votre configuration du lancement Auto Scaling avec la dernière AMI). Cette opération fournit une instance de conteneur propre avec les versions les plus récemment testées et validées d'Amazon Linux, de Docker, d'`ecs-init` et de l'agent de conteneur Amazon ECS. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.
- Connectez-vous à l'instance avec SSH et mettez à jour le package `ecs-init` (et ses dépendances) vers la dernière version. Cette opération fournit les versions les plus récemment testées et validées de Docker et d'`ecs-init` disponibles dans les référentiels Amazon Linux et la

dernière version de l'agent de conteneur Amazon ECS. Pour de plus amples informations, veuillez consulter Pour mettre à jour le package `ecs-init` sur une AMI optimisée pour Amazon ECS.

- Mettez à jour l'agent de conteneur avec l'opération de l'`UpdateContainerAgent`API, soit via la console, soit avec le AWS CLI ou AWS SDKs. Pour de plus amples informations, veuillez consulter Mise à jour de l'agent de conteneur Amazon ECS avec l'opération d'API `UpdateContainerAgent`.

> **Note**
>
> Les mises à jour de l'agent ne s'appliquent pas aux instances de conteneur Windows. Nous vous recommandons de lancer de nouvelles instances de conteneur pour mettre à jour le version de l'agent dans vos clusters Windows.

Pour mettre à jour le package **`ecs-init`** sur une AMI optimisée pour Amazon ECS

1. Connectez-vous à votre instance de conteneur via SSH.
2. Mettez à jour le package `ecs-init` avec la commande suivante.

```
sudo yum update -y ecs-init
```

> **Note**
>
> Le package `ecs-init` et l'agent de conteneur Amazon ECS sont mis à jour immédiatement. Cependant, les versions les plus récentes de Docker ne sont pas chargées tant que le démon Docker n'est pas redémarré. Redémarrez en relançant l'instance ou en exécutant les commandes suivantes sur votre instance :
>
> - AMI Amazon Linux 2 optimisée pour Amazon ECS :
>
> ```
> sudo systemctl restart docker
> ```
>
> - AMI Amazon Linux optimisée pour Amazon ECS :
>
> ```
> sudo service docker restart && sudo start ecs
> ```

Mise à jour de l'agent de conteneur Amazon ECS avec l'opération d'API **`UpdateContainerAgent`**

> **Important**
>
> L'API `UpdateContainerAgent` n'est prise en charge que sur les variantes Linux de l'AMI optimisée pour Amazon ECS, à l'exception de l'AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS. Pour les instances de conteneur utilisant l'AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS, mettez à jour le package `ecs-init` pour mettre à jour l'agent. Pour les instances de conteneur qui exécutent d'autres systèmes d'exploitation, consultez la page Mise à jour manuelle de l'agent de conteneur Amazon ECS (pour les applications non optimisées pour Amazon ECS AMIs). Si vous utilisez des instances de conteneur Windows, nous vous recommandons de lancer de nouvelles instances de conteneur pour mettre à jour le version de l'agent dans vos clusters Windows.

Le processus `UpdateContainerAgent` d'API commence lorsque vous demandez une mise à jour de l'agent, soit via la console, soit avec le AWS CLI ou AWS SDKs. Amazon ECS compare la version actuelle de votre agent à la dernière version d'agent disponible et vérifie si une mise à jour est possible. Si aucune mise à jour n'est disponible, si l'agent exécute déjà par exemple la version la plus récente, le message `NoUpdateAvailableException` est renvoyé.

Les étapes du processus de mise à jour ci-dessus sont les suivantes :

`PENDING`

Une mise à jour de l'agent est disponible, et le processus de mise à jour a commencé.

`STAGING`

L'agent a commencé le téléchargement de la mise à jour. Si l'agent ne peut pas télécharger la mise à jour, ou si le contenu de la mise à jour est incorrect ou endommagé, l'agent envoie une notification de l'échec et la mise à jour passe en état `FAILED`.

`STAGED`

Le téléchargement est terminé et le contenu de l'agent a été vérifié.

`UPDATING`

Le service `ecs-init` est redémarré et il récupère la nouvelle version de l'agent. Si, pour une raison quelconque, l'agent est incapable de redémarrer, la mise à jour passe à l'état `FAILED`. Dans le cas contraire, l'agent indique à Amazon ECS que la mise à jour est terminée.

> **Note**
>
> Les mises à jour de l'agent ne s'appliquent pas aux instances de conteneur Windows. Nous vous recommandons de lancer de nouvelles instances de conteneur pour mettre à jour le version de l'agent dans vos clusters Windows.

Pour mettre à jour l'agent de conteneur Amazon ECS sur une AMI optimisée pour Amazon ECS dans la console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, choisissez la région dans laquelle votre instance externe est inscrite.
3. Dans le panneau de navigation, choisissez Clusters et sélectionnez le cluster.
4. Sur la *name* page Cluster :, choisissez l'onglet Infrastructure.
5. Sous Instances de conteneur, sélectionnez les instances à mettre à jour, puis choisissez Actions et Mettre à jour l'agent.

Mise à jour manuelle de l'agent de conteneur Amazon ECS (pour les applications non optimisées pour Amazon ECS AMIs)

Il se peut que vous deviez parfois mettre à jour l'agent de conteneur Amazon ECS pour corriger des bogues et intégrer de nouvelles fonctionnalités. La mise à jour de l'agent de conteneur Amazon ECS n'interrompt pas les tâches ou les services en cours d'exécution sur l'instance de conteneur.

> **Note**
>
> Les mises à jour de l'agent ne s'appliquent pas aux instances de conteneur Windows. Nous vous recommandons de lancer de nouvelles instances de conteneur pour mettre à jour le version de l'agent dans vos clusters Windows.

1. Connectez-vous à votre instance de conteneur via SSH.
2. Vérifiez si votre agent utilise la variable d'environnement `ECS_DATADIR` pour enregistrer son état.

```
ubuntu:~$ docker inspect ecs-agent | grep ECS_DATADIR
```

Sortie :

```
"ECS_DATADIR=/data",
```

> **Important**
>
> Si la commande précédente ne renvoie pas la variable d'environnement `ECS_DATADIR`, vous devez arrêter toutes les tâches en cours d'exécution sur cette instance de conteneur avant de mettre à jour votre agent. Les agents récents avec la variable d'environnement `ECS_DATADIR` enregistrent leur état et sont possibles à mettre à jour sans problèmes tandis que les tâches sont en cours d'exécution.

3. Arrêtez l'agent de conteneur Amazon ECS.

```
ubuntu:~$ docker stop ecs-agent
```

4. Supprimez le conteneur de l'agent.

```
ubuntu:~$ docker rm ecs-agent
```

5. Vérifiez que le répertoire `/etc/ecs` et le fichier de configuration de l'agent de conteneur Amazon ECS existent à l'emplacement `/etc/ecs/ecs.config`.

```
ubuntu:~$ sudo mkdir -p /etc/ecs && sudo touch /etc/ecs/ecs.config
```

6. Modifiez le fichier `/etc/ecs/ecs.config` et vérifiez qu'il contient au moins les déclarations de variable suivantes. Pour que votre instance de conteneur ne s'inscrive pas auprès du cluster par défaut, spécifiez le nom de votre cluster comme valeur pour `ECS_CLUSTER`.

```
ECS_DATADIR=/data
ECS_ENABLE_TASK_IAM_ROLE=true
ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true
ECS_LOGFILE=/log/ecs-agent.log
ECS_AVAILABLE_LOGGING_DRIVERS=["json-file","awslogs"]
ECS_LOGLEVEL=info
ECS_CLUSTER=default
```

Pour plus d'informations sur ces options ainsi que d'autres options d'exécution d'agents, consultez la page Configuration de l'agent de conteneur Amazon ECS.

> **Note**
>
> Vous pouvez éventuellement stocker les variables d'environnement de votre agent dans Amazon S3 (qui peuvent être téléchargées sur vos instances de conteneur au moment du lancement à l'aide des données EC2 utilisateur Amazon). Ceci est particulièrement conseillé pour les informations sensibles, telles que les informations d'authentification pour les référentiels privés. Pour plus d'informations, consultez Stockage de la configuration de l'instance de conteneur Amazon ECS dans Amazon S3 et Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.

7. Extrayez la dernière image de l'agent de conteneur Amazon ECS depuis Amazon Elastic Container Registry Public.

```
ubuntu:~$ docker pull public.ecr.aws/ecs/amazon-ecs-agent:latest
```

Sortie :

```
Pulling repository amazon/amazon-ecs-agent
a5a56a5e13dc: Download complete
511136ea3c5a: Download complete
9950b5d678a1: Download complete
c48ddcf21b63: Download complete
Status: Image is up to date for amazon/amazon-ecs-agent:latest
```

8. Exécutez l'agent de conteneur Amazon ECS le plus récent sur votre instance de conteneur.

> **Note**
>
> Utilisez les stratégies de redémarrage de Docker ou un gestionnaire de processus (tel que upstart ou systemd) pour traiter l'agent de conteneur comme un service ou un démon, et vous assurer qu'il est redémarré après avoir été arrêté. L'AMI optimisée pour Amazon ECS utilise le `ecs-init` RPM à cette fin, et vous pouvez consulter le code source de ce RPM sur. GitHub

L'exemple suivant de commande d'exécution d'agent est divisé en lignes distinctes pour montrer chaque option. Pour plus d'informations sur ces options ainsi que d'autres options d'exécution d'agents, consultez la page Configuration de l'agent de conteneur Amazon ECS.

> **Important**
>
> Les systèmes d'exploitation sur lesquels cette `--privileged` option est SELinux activée doivent figurer dans votre docker run commande. En outre, pour les instances de conteneur SELinux activées, nous vous recommandons d'ajouter l'`:Z`option aux montages de `/data` volume `/log` et. Cependant, les montages hôtes de ces volumes doivent exister avant d'exécuter la commande. Dans le cas contraire, vous recevrez une erreur `no such file or directory`. Procédez comme suit si vous rencontrez des difficultés pour exécuter l'agent Amazon ECS sur une instance de conteneur SELinux activée :
>
> - Créez les points de montage des volumes hôtes sur votre instance de conteneur.
>
> ```
> ubuntu:~$ sudo mkdir -p /var/log/ecs /var/lib/ecs/data
> ```
>
> - Ajoutez l'option `--privileged` à la commande docker run ci-dessous.
> - Ajoutez l'option `:Z` aux montages de volume de conteneur `/log` et `/data` (par exemple, `--volume=/var/log/ecs/:/log:Z`) dans la commande docker run ci-dessous.

```
ubuntu:~$ sudo docker run --name ecs-agent \
--detach=true \
--restart=on-failure:10 \
--volume=/var/run:/var/run \
--volume=/var/log/ecs/:/log \
--volume=/var/lib/ecs/data:/data \
--volume=/etc/ecs:/etc/ecs \
--volume=/etc/ecs:/etc/ecs/pki \
--net=host \
--env-file=/etc/ecs/ecs.config \
amazon/amazon-ecs-agent:latest
```

> **Note**
>
> Si vous recevez un message `Error response from daemon: Cannot start container`, vous pouvez supprimer le conteneur ayant échoué à l'aide de la commande sudo docker rm ecs-agent, puis réessayer d'exécuter l'agent.

Windows optimisé pour Amazon ECS AMIs

Les modèles optimisés pour Amazon ECS AMIs sont préconfigurés avec les composants nécessaires dont vous avez besoin pour exécuter les charges de travail Amazon ECS. Bien que vous puissiez créer votre propre AMI d'instance de conteneur qui répond aux spécifications de base nécessaires pour exécuter vos charges de travail conteneurisées sur Amazon ECS, les applications optimisées pour Amazon ECS AMIs sont préconfigurées et testées sur Amazon ECS par des ingénieurs. AWS C'est la façon la plus simple de démarrer et d'exécuter rapidement vos conteneurs sur AWS .

Pour chaque variante, les métadonnées d'AMI optimisées pour Amazon ECS peuvent être récupérées par programmation. Ces métadonnées incluent le nom de l'AMI, la version de l'agent de conteneur Amazon ECS et la version d'exécution Amazon ECS qui inclut la version Docker. Pour de plus amples informations, veuillez consulter the section called “Récupération des métadonnées de l'AMI Windows optimisées pour Amazon ECS”.

> **Important**
>
> Toutes les variantes d'AMI optimisées pour l'ECS produites après août 2022 migreront de Docker EE (Mirantis) vers Docker CE (projet Moby).
> Pour garantir que les clients disposent des dernières mises à jour de sécurité par défaut, Amazon ECS gère au moins les trois dernières versions optimisées pour Windows Amazon ECS AMIs. Après avoir publié les nouvelles versions optimisées pour Windows Amazon ECS, AMIs Amazon ECS rend privées les versions optimisées pour Windows Amazon ECS AMIs qui sont plus anciennes. S'il existe une AMI privée à laquelle vous devez accéder, soumettez un ticket auprès du support Cloud.

Variantes d'AMI optimisées pour Amazon ECS

Les variantes Windows Server suivantes de l'AMI optimisée pour Amazon ECS sont disponibles pour vos instances Amazon EC2 .

> **Important**
>
> Toutes les variantes AMI optimisées pour ECS produites après le mois d'août migreront de Docker EE (Mirantis) vers Docker CE (projet Moby).

- AMI complète pour Windows Server 2025 optimisée pour Amazon ECS
- AMI principale pour Windows Server 2025 optimisée pour Amazon ECS
- AMI Windows Server 2022 Full optimisée pour Amazon ECS
- AMI de Windows Server 2022 Core optimisée pour Amazon ECS
- AMI Windows Server 2019 Full optimisée pour Amazon ECS
- AMI de Windows Server 2019 Core optimisée pour Amazon ECS
- AMI complète de Windows Server 2016 optimisée pour Amazon ECS

> **Important**
>
> Windows Server 2016 ne prend pas en charge la dernière version de Docker, par exemple 25.x.x. Par conséquent, Windows Server 2016 Full ne AMIs recevra pas de correctifs de sécurité ou de bogues pour le moteur d'exécution Docker. Nous vous recommandons de passer à l'une des plateformes Windows suivantes :
>
> - Windows Server 2022 Full
> - Windows Server 2022 Core
> - Windows Server 2019 Full
> - Windows Server 2019 Core

Le 9 août 2022, la date de fin de prise en charge de l'AMI Windows Server 20H2 Core optimisée pour Amazon ECS a été atteinte. Aucune nouvelle version de cette AMI ne sera publiée. Pour plus d'informations, consultez les informations sur la version de Windows Server.

Windows Server 2025, Windows Server 2022, Windows Server 2019 et Windows Server 2016 sont des versions LTSC (Long-Term Servicing Channel). Windows Server 20H2 est une version Canal semi-annuelle. Pour plus d'informations, consultez les informations sur la version de Windows Server.

Considérations

Voici quelques informations que vous devez savoir sur les conteneurs Amazon EC2 Windows et Amazon ECS.

- Les conteneurs Windows ne peuvent pas s'exécuter sur des instances de conteneur Linux, et inversement. Afin de garantir un meilleur placement de la tâche pour les tâches Windows et Linux, gardez les instances de conteneur Windows et Linux dans des clusters distincts et placez uniquement les tâches Windows sur des clusters Windows. Pour vous assurer que les définitions de tâche Windows sont uniquement placées sur des instances Windows, vous pouvez définir la contrainte de placement suivante : `memberOf(ecs.os-type=='windows')`.
- Les conteneurs Windows sont pris en charge pour les tâches utilisant les types de lancement Fargate EC2 et Fargate.
- Les conteneurs Windows et les instances de conteneur ne peuvent pas prendre en charge tous les paramètres de définition de tâche disponibles pour les conteneurs Linux et les instances de conteneur. Certains paramètres ne sont pas du tout pris en charge, tandis que d'autres se comportent différemment sous Windows et Linux. Pour de plus amples informations, veuillez consulter Différences de définition des tâches Amazon ECS pour EC2 les instances exécutant Windows.
- Pour les rôles IAM pour la fonction des tâches, vous devez configurer vos instances de conteneur Windows pour autoriser la fonction lors du lancement. Vos conteneurs doivent exécuter le PowerShell code fourni lorsqu'ils utilisent cette fonctionnalité. Pour de plus amples informations, veuillez consulter Configuration supplémentaire de l'instance Amazon EC2 Windows.
- Les rôles IAM pour la fonction des tâches utilisent un proxy d'informations d'identification pour fournir des informations d'identification aux conteneurs. Ce proxy d'informations d'identification occupe le port 80 sur l'instance de conteneur, donc si vous utilisez des rôles IAM pour les tâches, le port 80 n'est pas disponible pour les tâches. Pour les conteneurs de service web, vous pouvez utiliser un Application Load Balancer et un mappage de port dynamique pour fournir des connexions port 80 HTTP standard à vos conteneurs. Pour de plus amples informations, veuillez consulter Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS.
- Les images Docker de Windows Server sont volumineuses (9 Gio). Ainsi, vos instances de conteneur Windows nécessitent plus d'espace de stockage que les instances de conteneur Linux.
- Pour exécuter un conteneur Windows sur un Windows Server, la version du système d'exploitation de l'image de base du conteneur doit correspondre à celle de l'hôte. Pour plus d'informations, consultez Compatibilité avec la version du conteneur Windows sur le site web de documentation Microsoft. Si votre cluster exécute plusieurs versions de Windows, vous

pouvez vous assurer qu'une tâche est placée sur une EC2 instance exécutée sur la même version en utilisant la contrainte de placement :`memberOf(attribute:ecs.os-family == WINDOWS_SERVER_<OS_Release>_<FULL or CORE>)`. Pour de plus amples informations, veuillez consulter the section called “Récupération des métadonnées de l'AMI Windows optimisées pour Amazon ECS”.

Récupération des métadonnées de l'AMI Windows optimisées pour Amazon ECS

L'ID de l'AMI, le nom de l'image, le système d'exploitation, la version de l'agent de conteneur et la version d'exécution de chaque variante d'Amazon ECS-Optimized AMIs peuvent être récupérés par programmation en interrogeant l'API du magasin de paramètres de Systems Manager. Pour plus d'informations sur l'API Systems Manager Parameter Store, reportez-vous aux sections GetParameterset GetParametersByPath.

> **Note**
>
> Votre compte administratif doit avoir les autorisations IAM suivantes pour extraire les métadonnées d'AMI optimisée pour Amazon ECS. Ces autorisations ont été ajoutées à la politique IAM `AmazonECS_FullAccess`.
>
> - SMS : GetParameters
> - SMS : GetParameter
> - SMS : GetParametersByPath

Format de paramètre Systems Manager Parameter Store

> **Note**
>
> Les paramètres de l'API Systems Manager Parameter Store suivants sont obsolètes et ne doivent pas être utilisés pour récupérer la dernière version de Windows : AMIs
>
> - `/aws/service/ecs/optimized-ami/windows_server/2016/english/full/recommended/image_id`
> - `/aws/service/ecs/optimized-ami/windows_server/2019/english/full/recommended/image_id`

Les informations ci-dessous présentent le format de nom de paramètre pour chaque variante d'AMI optimisée pour Amazon ECS.

- Métadonnées AMI complètes pour Windows Server 2025 :

```
/aws/service/ami-windows-latest/Windows_Server-2025-English-Full-ECS_Optimized
```

- Métadonnées de l'AMI Windows Server 2025 Core :

```
/aws/service/ami-windows-latest/Windows_Server-2025-English-Core-ECS_Optimized
```

- Métadonnées de l'AMI Windows Server 2022 Full :

```
/aws/service/ami-windows-latest/Windows_Server-2022-English-Full-ECS_Optimized
```

- Métadonnées de l'AMI Windows Server 2022 Core :

```
/aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized
```

- Métadonnées de l'AMI Windows Server 2019 Full :

```
/aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized
```

- Métadonnées de l'AMI principale de Windows Server 2019 :

```
/aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized
```

- Métadonnées de l'AMI complète de Windows Server 2016 :

```
/aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized
```

Le format de nom de paramètre suivant extrait les métadonnées de la dernière AMI de base complète Windows Server 2019.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-
English-Full-ECS_Optimized
```

Les informations suivantes présentent un exemple de l'objet JSON renvoyé pour la valeur du paramètre.

```
{
    "Parameters": [
        {
            "Name": "/aws/service/ami-windows-latest/Windows_Server-2019-English-Full-
ECS_Optimized",
            "Type": "String",
            "Value": "{\"image_name\":\"Windows_Server-2019-English-Full-
ECS_Optimized-2023.06.13\",\"image_id\":\"ami-0debc1fb48e4aee16\",\"ecs_runtime_version
\":\"Docker (CE) version 20.10.21\",\"ecs_agent_version\":\"1.72.0\"}",
            "Version": 58,
            "LastModifiedDate": "2023-06-22T19:37:37.841000-04:00",
            "ARN": "arn:aws:ssm:us-east-1::parameter/aws/service/ami-windows-latest/
Windows_Server-2019-English-Full-ECS_Optimized",
            "DataType": "text"
        }
    ],
    "InvalidParameters": []
}
```

Chacun des champs dans la sortie obtenue ci-dessus est disponible pour être interrogé comme sous-paramètres. Construisez le chemin d'accès d'un sous-paramètre en ajoutant le nom du sous-paramètre au chemin d'accès de l'AMI sélectionnée. Les sous-paramètres suivants sont disponibles :

- `schema_version`
- `image_id`
- `image_name`
- `os`
- `ecs_agent_version`
- `ecs_runtime_version`

Exemples

Les exemples suivants montrent comment vous pouvez récupérer les métadonnées de chaque variante d'AMI optimisée pour Amazon ECS.

Extraction des métadonnées de la dernière AMI optimisée pour Amazon ECS stable

Vous pouvez récupérer la dernière AMI stable optimisée pour Amazon ECS à l' AWS CLI aide des commandes suivantes AWS CLI .

- Pour l'AMI complète Windows Server 2025 optimisée pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-
English-Full-ECS_Optimized --region us-east-1
```

- Pour l'AMI principale Windows Server 2025 optimisée pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-
English-Core-ECS_Optimized --region us-east-1
```

- Pour l'AMI Windows Server 2022 Full optimisée pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-
English-Full-ECS_Optimized --region us-east-1
```

- Pour l'AMI Windows Server 2022 optimisée pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-
English-Core-ECS_Optimized --region us-east-1
```

- Pour l'AMI Windows Server 2019 Full optimisée pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-
English-Full-ECS_Optimized --region us-east-1
```

- Pour l'AMI Windows Server 2019 Core optimisée pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-
English-Core-ECS_Optimized --region us-east-1
```

- Pour l'AMI Windows Server 2016 Full optimisée pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2016-
English-Full-ECS_Optimized --region us-east-1
```

Utilisation de l'AMI optimisée pour Amazon ECS la plus récente recommandée dans un modèle AWS CloudFormation

Vous pouvez référencer la dernière AMI optimisée pour Amazon ECS recommandée dans un modèle AWS CloudFormation en référençant le nom du magasin de paramètres Systems Manager.

```
Parameters:
  LatestECSOptimizedAMI:
    Description: AMI ID
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-
ECS_Optimized/image_id
```

Versions de l'AMI Windows optimisées pour Amazon ECS

Consultez les versions actuelles et précédentes de l'outil optimisé pour Amazon ECS AMIs et les versions correspondantes de l'agent de conteneur Amazon ECS, de Docker et du package. `ecs-init`

Les métadonnées d'AMI optimisée pour Amazon ECS, notamment l'ID d'AMI, peuvent être extraites par programmation pour chaque variante. Pour de plus amples informations, veuillez consulter the section called “Récupération des métadonnées de l'AMI Windows optimisées pour Amazon ECS”.

Les onglets suivants affichent la liste des versions optimisées pour Windows Amazon ECS AMIs . Pour plus de détails sur le référencement du paramètre Systems Manager Parameter Store dans un AWS CloudFormation modèle, consultezUtilisation de l'AMI optimisée pour Amazon ECS la plus récente recommandée dans un modèle AWS CloudFormation.

> **Important**
>
> Pour garantir que les clients disposent des dernières mises à jour de sécurité par défaut, Amazon ECS gère au moins les trois dernières versions optimisées pour Windows Amazon ECS AMIs. Après avoir publié les nouvelles versions optimisées pour Windows Amazon ECS, AMIs Amazon ECS rend privées les versions optimisées pour Windows Amazon ECS AMIs qui sont plus anciennes. S'il existe une AMI privée à laquelle vous devez accéder, soumettez un ticket auprès du support Cloud.
> Windows Server 2016 ne prend pas en charge la dernière version de Docker, par exemple 25.x.x. Par conséquent, Windows Server 2016 Full ne AMIs recevra pas de correctifs de sécurité ou de bogues pour le moteur d'exécution Docker. Nous vous recommandons de passer à l'une des plateformes Windows suivantes :
>
> - Windows Server 2022 Full
> - Windows Server 2022 Core
> - Windows Server 2019 Full

- Windows Server 2019 Core

Windows Server 2025 Full AMI versions

Le tableau ci-dessous répertorie les versions actuelles et précédentes de l'AMI complète Windows Server 2025 optimisée pour Amazon ECS, ainsi que les versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

AMI complète pour Windows Server 2025 optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2025-English-Full-ECS_Optimized-2025.06.13	`1.94.0`	`25.0.6 (Docker CE)`	Public

Utilisez la AWS CLI commande suivante pour récupérer l'AMI complète Windows Server 2025 actuellement optimisée pour Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-English-Full-ECS_Optimized
```

Windows Server 2025 Core AMI versions

Le tableau ci-dessous répertorie les versions actuelles et précédentes de l'AMI Windows Server 2025 Core optimisée pour Amazon ECS, ainsi que les versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

AMI principale pour Windows Server 2025 optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2025-English-Core-ECS_Optimized-2025.06.13	`1.94.0`	`25.0.6 (Docker CE)`	Public

Utilisez la AWS CLI commande suivante pour récupérer l'AMI principale Windows Server 2025 actuellement optimisée pour Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-English-Core-ECS_Optimized
```

Windows Server 2022 Full AMI versions

Le tableau ci-dessous répertorie les versions actuelles et antérieures de l'AMI Windows Server 2022 Full optimisée pour Amazon ECS, ainsi que les versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

AMI Windows Server 2022 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2_English-Full-ECS_Optimized-2025.06.13	`1.94.0`	`25.0.6 (Docker CE)`	Public
Windows_Server-2_English-Full-ECS_Optimized-2025.05.17	`1.93.1`	`25.0.6 (Docker CE)`	Public
Windows_Server-2_English-Full-ECS_Optimized-2025.4.14	`1.91.2`	`25.0.6 (Docker CE)`	Public

AMI Windows Server 2022 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2_English-Full-ECS_Optimized-2025.3.17	1.91.0	25.0.6 (Docker CE)	Public
Windows_Server-2_English-Full-ECS_Optimized-2025.2.14	1.90.0	25.0.6 (Docker CE)	Public

Utilisez la AWS CLI commande suivante pour récupérer l'AMI complète Windows Server 2022 actuellement optimisée pour Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Full-ECS_Optimized
```

Windows Server 2022 Core AMI versions

Le tableau ci-dessous répertorie les versions actuelles et antérieures de l'AMI Windows Server 2022 Core optimisée pour Amazon ECS, ainsi que les versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

AMI de Windows Server 2022 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2_English-Core-ECS_Optimized-2025.06.13	1.94.0	25.0.6 (Docker CE)	Public
Windows_Server-2_English-Core-ECS_Optimized-2025.05.17	1.93.1	25.0.6 (Docker CE)	Public

AMI de Windows Server 2022 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2_English-Core-ECS_Optimized-2025.4.14	`1.91.2`	`25.0.6 (Docker CE)`	Public
Windows_Server-2_English-Core-ECS_Optimized-2025.3.17	`1.91.0`	`25.0.6 (Docker CE)`	Public
Windows_Server-2_English-Core-ECS_Optimized-2025.2.14	`1.90.0`	`25.0.6 (Docker CE)`	Public

Utilisez la AWS CLI commande suivante pour récupérer l'AMI complète Windows Server 2022 actuellement optimisée pour Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized
```

Windows Server 2019 Full AMI versions

Le tableau ci-dessous répertorie les versions actuelles et antérieures de l'AMI Windows Server 2019 Full optimisée pour Amazon ECS, ainsi que les versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

AMI Windows Server 2019 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-Englis-Full-ECS_Optimized-2025.06.13	`1.94.0`	`25.0.6 (Docker CE)`	Public

AMI Windows Server 2019 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-Englis-Full-ECS_Optimized-2025.05.17	`1.93.1`	`25.0.6 (Docker CE)`	Public
Windows_Server-2019-Englis-Full-ECS_Optimized-2025.4.14	`1.91.2`	`25.0.6 (Docker CE)`	Public
Windows_Server-2019-Englis-Full-ECS_Optimized-2025.3.17	`1.91.0`	`25.0.6 (Docker CE)`	Public
Windows_Server-2019-Englis-Full-ECS_Optimized-2025.2.14	`1.90.0`	`25.0.6 (Docker CE)`	Public

Utilisez la AWS CLI commande suivante pour récupérer l'AMI complète Windows Server 2019 actuellement optimisée pour Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized
```

Windows Server 2019 Core AMI versions

Le tableau ci-dessous répertorie les versions actuelles et antérieures de l'AMI Windows Server 2019 Core optimisée pour Amazon ECS, ainsi que les versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

AMI de Windows Server 2019 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-Englis-Core-ECS_Optimized-2025.06.13	`1.94.0`	`25.0.6 (Docker CE)`	Public
Windows_Server-2019-Englis-Core-ECS_Optimized-2025.05.17	`1.93.1`	`25.0.6 (Docker CE)`	Public
Windows_Server-2019-Englis-Core-ECS_Optimized-2025.4.14	`1.91.2`	`25.0.6 (Docker CE)`	Public
Windows_Server-2019-Englis-Core-ECS_Optimized-2025.3.17	`1.91.0`	`25.0.6 (Docker CE)`	Public
Windows_Server-2019-Englis-Core-ECS_Optimized-2025.2.14	`1.90.0`	`25.0.6 (Docker CE)`	Public

Utilisez la AWS CLI commande suivante pour récupérer l'AMI complète Windows Server 2019 actuellement optimisée pour Amazon ECS.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized
```

Windows Server 2016 Full AMI versions

> **Important**
>
> Windows Server 2016 ne prend pas en charge la dernière version de Docker, par exemple 25.x.x. Par conséquent, Windows Server 2016 Full ne AMIs recevra pas de correctifs de sécurité ou de bogues pour le moteur d'exécution Docker. Nous vous recommandons de passer à l'une des plateformes Windows suivantes :
>
> - Windows Server 2022 Full
> - Windows Server 2022 Core
> - Windows Server 2019 Full
> - Windows Server 2019 Core

Le tableau ci-dessous répertorie les versions actuelles et antérieures de l'AMI Windows Server 2016 Full optimisée pour Amazon ECS, ainsi que les versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

AMI complète de Windows Server 2016 optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2016-English-Full-ECS_Optimized-2025.06.13	`1.94.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS_Optimized-2025.05.17	`1.93.1`	`20.10.23 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS	`1.91.2`	`20.10.23 (Docker CE)`	Public

AMI complète de Windows Server 2016 optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
_Optimized-2025.4.14			
Windows_Server-2016-English-Full-ECS_Optimized-2025.3.17	`1.91.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS_Optimized-2025.2.14	`1.90.0`	`20.10.23 (Docker CE)`	Public

Utilisez l'AMI complète Windows Server 2016 optimisée pour AWS CLI Amazon ECS suivante.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized
```

Création de votre propre AMI Windows optimisée pour Amazon ECS

Utilisez EC2 Image Builder pour créer votre propre AMI Windows personnalisée optimisée pour Amazon ECS. Cela facilite l'utilisation d'une AMI Windows avec votre propre licence sur Amazon ECS. Amazon ECS propose un composant Image Builder géré qui fournit la configuration système nécessaire pour exécuter des instances Windows afin d'héberger vos conteneurs. Chaque composant géré par Amazon ECS comprend un agent de conteneur spécifique et une version Docker. Vous pouvez personnaliser votre image pour utiliser le dernier composant géré Amazon ECS, ou si un agent de conteneur plus ancien ou une version Docker plus ancienne est nécessaire, vous pouvez spécifier un autre composant.

Pour une présentation complète de l'utilisation d' EC2 Image Builder, voir Getting started with EC2 Image Builder dans le guide de l'utilisateur d'EC2 Image Builder.

Lorsque vous créez votre propre AMI Windows optimisée pour Amazon ECS à l'aide d' EC2 Image Builder, vous créez une recette d'image. Votre recette d'image doit répondre aux critères suivants :

- L'image source doit être basée sur Windows Server 2019 Core, Windows Server 2019 Full, Windows Server 2022 Core ou Windows Server 2022 Full. Tout autre système d'exploitation Windows n'est pas pris en charge et peut ne pas être compatible avec le composant.
- Lorsque vous spécifiez les composants de création, le composant `ecs-optimized-ami-windows` est requis. Le composant `update-windows` est recommandé, ce qui garantit que l'image contient les dernières mises à jour de sécurité.

  Pour spécifier une version de composant différente, développez le menu Versioning options (Options de gestion des versions) et spécifiez la version du composant à utiliser. Pour de plus amples informations, veuillez consulter Établissement de la liste des versions de composants `ecs-optimized-ami-windows`.

Établissement de la liste des versions de composants **`ecs-optimized-ami-windows`**

Lorsque vous créez une recette EC2 Image Builder et que vous spécifiez le `ecs-optimized-ami-windows` composant, vous pouvez soit utiliser l'option par défaut, soit spécifier une version de composant spécifique. Pour déterminer les versions de composants disponibles, ainsi que l'agent de conteneur Amazon ECS et les versions Docker contenues dans le composant, vous pouvez utiliser la AWS Management Console.

Pour répertorier les versions de composants **`ecs-optimized-ami-windows`** disponibles

1. Ouvrez la console EC2 Image Builder à l'adresse https://console.aws.amazon.com/imagebuilder/.
2. Dans la barre de navigation, sélectionnez la région dans laquelle votre image est créée.
3. Dans le panneau de navigation, sous le menu Saved configurations (Configurations enregistrées), choisissez Components (Composants).
4. Sur la page Components (Composants), dans la barre de recherche, saisissez `ecs-optimized-ami-windows` et faites défiler le menu de qualification et sélectionnez Quick start (Amazon-managed) [Quick Start (géré par Amazon)].
5. Utilisation de la colonne Description pour déterminer la version du composant avec l'agent de conteneur Amazon ECS et la version Docker qu'exige votre image.

Gestion des instances de conteneurs Windows Amazon ECS

Lorsque vous utilisez EC2 des instances pour vos charges de travail Amazon ECS, vous êtes responsable de leur maintenance.

Les mises à jour de l'agent ne s'appliquent pas aux instances de conteneur Windows. Nous vous recommandons de lancer de nouvelles instances de conteneur pour mettre à jour le version de l'agent dans vos clusters Windows.

Procédures de gestion

- Lancement d'une instance de conteneur Amazon ECS Windows
- Démarrage des instances de conteneur Windows Amazon ECS pour transmettre des données
- Utilisation d'un proxy HTTP pour les instances de conteneur Windows Amazon ECS
- Configuration des instances de conteneur Windows Amazon ECS pour recevoir des notifications relatives aux instances Spot

Lancement d'une instance de conteneur Amazon ECS Windows

Vos instances de conteneur Amazon ECS sont créées à l'aide de la EC2 console Amazon. Avant de commencer, assurez-vous d'avoir terminé les étapes de Configurer l'utilisation d'Amazon ECS.

Pour plus d'informations sur l'assistant de lancement, consultez Lancer une instance à l'aide du nouvel assistant de lancement d'instance dans le guide de EC2 l'utilisateur Amazon.

Vous pouvez utiliser le nouvel EC2 assistant Amazon pour lancer une instance. Vous pouvez utiliser la liste suivante pour les paramètres et laisser les paramètres non répertoriés comme paramètres par défaut. Les instructions suivantes vous guident dans chaque groupe de paramètres.

Procédure

Avant de commencer, complétez les étapes détaillées dans Configurer l'utilisation d'Amazon ECS.

1. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans la barre de navigation en haut de l'écran, la AWS région actuelle est affichée (par exemple, USA East (Ohio)). Sélectionnez une région dans laquelle lancer l'instance. Ce choix est important car certaines EC2 ressources Amazon peuvent être partagées entre les régions, alors que d'autres ne le peuvent pas.
3. Dans le tableau de bord de EC2 la console Amazon, choisissez Launch instance.

Noms et identifications

Le nom de l'instance est une identification, où la clé est Name (Nom), et la valeur est le nom que vous spécifiez. Vous pouvez étiqueter l'instance, les volumes et les Elastic Graphics. Pour les instances Spot, vous pouvez baliser uniquement la demande d'instance Spot.

La spécification d'un nom d'instance et d'identifications supplémentaires est facultative.

- Pour Name (Nom), saisissez un nom descriptif pour l'instance. Si vous ne spécifiez pas de nom, l'instance peut être identifiée par son ID, qui est automatiquement généré lorsque vous lancez l'instance.
- Pour ajouter des identifications supplémentaires, sélectionnez Add additional tags (Ajouter des identifications supplémentaires). Choisissez Add tag (Ajouter une identification), saisissez une clé et une valeur, puis sélectionnez le type de ressource à étiqueter. Choisissez Add tag (Ajouter une identification) pour chaque étiquette supplémentaire.

Images d'applications et de systèmes d'exploitation (Amazon Machine Image)

Une Amazon Machine Image (AMI) contient les informations requises pour créer une instance. Par exemple, une AMI peut contenir le logiciel nécessaire pour fonctionner en tant que serveur web, comme Apache et votre site web.

Pour connaître les dernières versions optimisées pour Amazon ECS AMIs et leurs valeurs, consultez l'AMI optimisée pour Windows Amazon ECS.

Utilisez la barre de recherche pour trouver une AMI optimisée pour Amazon ECS adaptée publiée par. AWS

1. En fonction de vos besoins, saisissez l'une des options suivantes AMIs dans la barre de recherche et appuyez sur Entrée.
   - Windows_Server-2022-English-Full-ECS_Optimized
   - Windows_Server-2022-English-Core-ECS_Optimized
   - Windows_Server-2019-English-Full-ECS_Optimized
   - Windows_Server-2019-English-Core-ECS_Optimized
   - Windows_Server-2016-English-Full-ECS_Optimized
2. Sur la page Choose an Amazon Machine Image (AMI), sélectionnez l' AMIsonglet Communauté.
3. Dans la liste qui apparaît, choisissez une AMI vérifiée par Microsoft avec la date de publication la plus récente et cliquez sur Sélectionner.

Type d'instance

Le type d'instance définit la configuration matérielle et la taille de l'instance. Les types d'instance plus importants disposent de plus d'UC et de mémoire. Pour plus d'informations, consultez Types d'instance.

- Pour Instance type (Type d'instance), sélectionnez le type de l'instance.

  Le type d'instance que vous sélectionnez détermine les ressources disponibles pour les tâches à exécuter.

Paire de clés (connexion)

Pour Key pair name (Nom de la paire de clés), choisissez une paire de clés existante ou choisissez Create new key pair (Créer une nouvelle paire de clés) pour en créer une nouvelle.

> **Important**
>
> Si vous sélectionnez l'option Proceed without key pair (Not recommended) ((Continuer sans paire de clé) (Non recommandé)), vous ne pourrez pas vous connecter à l'instance à moins de choisir une AMI configurée de façon à autoriser les utilisateurs à se connecter d'une autre façon.

Paramètres réseau

Configurez les paramètres réseau, le cas échéant.

- Networking platform (Plateforme de mise en réseau) : choisissez Virtual Private Cloud (VPC) (Cloud privé virtuel [VPC]), puis spécifiez le sous-réseau dans la section Network interfaces (Interfaces réseau).
- VPC : sélectionnez un VPC existant dans lequel créer le groupe de sécurité.
- Sous-réseau : vous pouvez lancer une instance dans un sous-réseau associé à une zone de disponibilité, une zone locale, une zone Wavelength ou un Outpost.

  Pour lancer l'instance dans une zone de disponibilité, sélectionnez le sous-réseau dans lequel lancer votre instance. Pour créer un sous-réseau, choisissez Créer un nouveau sous-réseau afin d'accéder à la console Amazon VPC. Une fois que vous avez terminé, revenez dans l'assistant de

lancement d'instance et choisissez l'icône Refresh (Actualiser) afin de charger votre sous-réseau dans la liste.

Pour lancer l'instance dans une zone locale, sélectionnez un sous-réseau que vous avez créé dans la zone locale.

Pour lancer une instance dans un Outpost, sélectionnez un sous-réseau dans un VPC que vous avez associé à l'Outpost.

- Auto-assign Public IP (Attribuer automatiquement l'adresse IP publique) : si votre instance doit être accessible à partir d'Internet, vérifiez que le champ Auto-assign Public IP (Attribuer automatiquement l'adresse IP publique) est défini sur Enable (Activer). Si ce n'est pas le cas, définissez ce champ sur Disable (Désactiver).

> **Note**
>
> Les instances de conteneur ont besoin de communiquer avec le point de terminaison de service Amazon ECS. Cela peut être via un point de terminaison d'un VPC d'interface ou via vos instances de conteneur ayant des adresses IP publiques.
> Pour plus d'informations sur les points de terminaison d'un VPC d'interface, consultez Points de terminaison d'un VPC d'interface Amazon ECS (AWS PrivateLink).
> Si vous n'avez pas de point de terminaison d'un VPC d'interface configuré et que vos instances de conteneur n'ont pas d'adresses IP publiques, elles doivent utiliser la traduction d'adresses réseau (NAT) pour fournir cet accès. Pour de plus amples informations, veuillez consulter Passerelles NAT dans le Guide de l'utilisateur Amazon VPC et Utilisation d'un proxy HTTP pour les instances de conteneur Amazon ECS Linux dans ce guide.

- Firewall (security groups) (Pare-feu (groupes de sécurité)) : utilisez un groupe de sécurité afin de définir les règles de pare-feu de votre instance de conteneur. Ces règles déterminent le trafic réseau entrant acheminé vers votre instance de conteneur. Le reste du trafic est ignoré.
  - Pour sélectionner un groupe de sécurité existant, choisissez Select existing security group (Sélectionner un groupe de sécurité existant), puis sélectionnez le groupe de sécurité que vous avez créé dans Configurer l'utilisation d'Amazon ECS.

Configurer le stockage

L'AMI sélectionnée inclut un ou plusieurs volumes de stockage, notamment le volume racine. Vous pouvez spécifier d'autres volumes à attacher à l'instance.

Vous pouvez utiliser la vue Simple.

- Storage type (Type de stockage) : configurez le stockage pour votre instance de conteneur.

  Si vous utilisez l'AMI Amazon Linux optimisée pour Amazon ECS, deux volumes sont configurés sur votre instance. Le volume Racine est réservé au système d'exploitation et le second volume Amazon EBS (attaché à `/dev/xvdcz`) est réservé à Docker.

  Vous pouvez augmenter ou diminuer la taille des volumes pour votre instance afin de répondre aux besoins de votre application.

Détails avancés

Développez la section Détails avancés pour afficher les champs et spécifier des paramètres supplémentaires pour l'instance.

- Purchasing option (Option d'achat) : sélectionnez Request Spot Instances (Demander des instances Spot) pour demander une instance Spot. Vous devez également définir les autres champs associés aux instances Spot. Pour plus d'informations, consultez Demandes d'instances Spot.

  > **Note**
  >
  > Si vous utilisez les instances Spot et qu'un message `Not available` s'affiche, vous devrez peut-être choisir un autre type d'instance.

  .

- IAM instance profile (Profil d'instance IAM) : sélectionnez votre rôle IAM d'instance de conteneur. Celui-ci est généralement nommé `ecsInstanceRole`.

  > **Important**
  >
  > Si vous ne pas lancez votre instance de conteneur avec les autorisations IAM appropriées, votre agent Amazon ECS ne peut pas se connecter à votre cluster. Pour de plus amples informations, veuillez consulter Rôle IAM d'instance de conteneur Amazon ECS.

- (Facultatif) User data (Données utilisateur) : configurez votre instance de conteneur Amazon ECS avec les données utilisateur, telles que les variables d'environnement d'agent depuis Configuration

de l'agent de conteneur Amazon ECS. Les scripts de données EC2 utilisateur Amazon ne sont exécutés qu'une seule fois, lors du premier lancement de l'instance. Les exemples suivants présentent des utilisations courantes des données utilisateur :

- Par défaut, votre instance de conteneur est lancée dans votre cluster par défaut. Pour un lancement dans un cluster autre que celui défini par défaut, choisissez la liste Détails avancés. Collez ensuite le script suivant dans le champ Données utilisateur, en le *your_cluster_name* remplaçant par le nom de votre cluster.

  `EnableTaskIAMRole` active la fonction Rôles IAM pour les tâches.

  En outre, les options suivantes sont disponibles lorsque vous utilisez le mode réseau `awsvpc`.

  - `EnableTaskENI` : cet indicateur active les réseaux des tâches et est requis lorsque vous utilisez le mode réseau `awsvpc`.
  - `AwsvpcBlockIMDS` : cet indicateur facultatif bloque l'accès IMDS pour les conteneurs de tâches qui s'exécutent dans en mode réseau `awsvpc`.
  - `AwsvpcAdditionalLocalRoutes` : cet indicateur facultatif vous permet d'avoir des acheminements supplémentaires dans l'espace de noms de tâche.

    Remplacez `ip-address` par l'adresse IP pour les acheminements supplémentaires, par exemple 172.31.42.23/32.

```
<powershell>
Import-Module ECSTools
Initialize-ECSAgent -Cluster your_cluster_name -EnableTaskIAMRole -EnableTaskENI -
AwsvpcBlockIMDS -AwsvpcAdditionalLocalRoutes
'["ip-address"]'
</powershell>
```

Démarrage des instances de conteneur Windows Amazon ECS pour transmettre des données

Lorsque vous lancez une EC2 instance Amazon, vous pouvez lui transmettre des EC2 données utilisateur. Ces données peuvent être utilisées afin d'effectuer des tâches de configuration automatisées courantes, voire d'exécuter des scripts lors du démarrage de l'instance. Pour Amazon ECS, les données utilisateur sont le plus souvent utilisées pour transmettre les informations de configuration au démon Docker et à l'agent de conteneur Amazon ECS.

Vous pouvez transmettre plusieurs types de données utilisateur à Amazon EC2, notamment des boothooks cloud, des scripts shell et `cloud-init` des directives. Pour plus d'informations sur ce point et sur d'autres types de format, consultez la documentation sur Cloud-Init.

Vous pouvez transmettre ces données utilisateur lorsque vous utilisez l'assistant de EC2 lancement Amazon. Pour de plus amples informations, veuillez consulter Lancement d'une instance de conteneur Amazon ECS Linux.

Données utilisateur Windows par défaut

Cet exemple de script de données utilisateur montre les données utilisateur par défaut que vos instances de conteneur Windows reçoivent si vous utilisez la console. Le script ci-dessous :

- Définit le nom du cluster selon le nom que vous avez saisi.
- Définit les rôles IAM pour les tâches.
- Définit `json-file` et `awslogs` comme pilotes de journalisation disponibles.

En outre, les options suivantes sont disponibles lorsque vous utilisez le mode réseau `awsvpc`.

- `EnableTaskENI` : cet indicateur active les réseaux des tâches et est requis lorsque vous utilisez le mode réseau `awsvpc`.
- `AwsvpcBlockIMDS` : cet indicateur facultatif bloque l'accès IMDS pour les conteneurs de tâches qui s'exécutent dans en mode réseau `awsvpc`.
- `AwsvpcAdditionalLocalRoutes` : cet indicateur facultatif vous permet d'avoir des acheminements supplémentaires.

  Remplacez `ip-address` par l'adresse IP pour les acheminements supplémentaires, par exemple 172.31.42.23/32.

Vous pouvez utiliser ce script pour vos propres instances de conteneur (sous réserve qu'elles soient lancées à partir d'une AMI Windows Server optimisée pour Amazon ECS).

Remplacez la ligne `-Cluster` *`cluster-name`* pour spécifier votre propre nom de cluster.

```
<powershell>
Initialize-ECSAgent -Cluster cluster-name -EnableTaskIAMRole -LoggingDrivers '["json-
file","awslogs"]' -EnableTaskENI -AwsvpcBlockIMDS -AwsvpcAdditionalLocalRoutes
'["ip-address"]'
```

```
</powershell>
```

Pour les tâches Windows configurées pour utiliser le pilote de journalisation `awslogs`, vous devez également définir la variable d'environnement `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE` sur votre instance de conteneur. Utilisez la syntaxe suivante.

Remplacez la ligne `-Cluster` *`cluster-name`* pour spécifier votre propre nom de cluster.

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster cluster-name -EnableTaskIAMRole -LoggingDrivers '["json-
file","awslogs"]'
</powershell>
```

Données utilisateur d'installation d'un agent Windows

Cet exemple de script de données utilisateur installe l'agent de conteneur Amazon ECS sur une instance lancée avec une AMI Windows_Server-2016-English-Full-Containers. Il a été adapté à partir des instructions d'installation de l'agent figurant sur la page README du GitHubréférentiel Amazon ECS Container Agent.

> **Note**
>
> Ce script est partagé à titre d'exemple. Il est beaucoup plus facile de commencer avec les conteneurs Windows en utilisant l'AMI Windows Server optimisée pour Amazon ECS. Pour de plus amples informations, veuillez consulter Création d'un cluster Amazon ECS pour le type de lancement Fargate.

Pour plus d'informations sur l'installation de l'agent Amazon ECS sur Windows Server 2022 Full, consultez le numéro 3753 sur GitHub.

Vous pouvez utiliser ce script pour vos propres instances de conteneur (à condition qu'elles soient lancées avec une version de l'AMI Windows_Server-2016-English-Full-Containers). Veillez à modifier la ligne *`windows`* dans le fichier de configuration en spécifiant votre propre nom de cluster (si vous n'utilisez pas un cluster nommé `windows`).

```
<powershell>
```

```
# Set up directories the agent uses
New-Item -Type directory -Path ${env:ProgramFiles}\Amazon\ECS -Force
New-Item -Type directory -Path ${env:ProgramData}\Amazon\ECS -Force
New-Item -Type directory -Path ${env:ProgramData}\Amazon\ECS\data -Force
# Set up configuration
$ecsExeDir = "${env:ProgramFiles}\Amazon\ECS"
[Environment]::SetEnvironmentVariable("ECS_CLUSTER", "windows", "Machine")
[Environment]::SetEnvironmentVariable("ECS_LOGFILE", "${env:ProgramData}\Amazon\ECS\log
\ecs-agent.log", "Machine")
[Environment]::SetEnvironmentVariable("ECS_DATADIR", "${env:ProgramData}\Amazon\ECS
\data", "Machine")
# Download the agent
$agentVersion = "latest"
$agentZipUri = "https://s3.amazonaws.com/amazon-ecs-agent/ecs-agent-windows-
$agentVersion.zip"
$zipFile = "${env:TEMP}\ecs-agent.zip"
Invoke-RestMethod -OutFile $zipFile -Uri $agentZipUri
# Put the executables in the executable directory.
Expand-Archive -Path $zipFile -DestinationPath $ecsExeDir -Force
Set-Location ${ecsExeDir}
# Set $EnableTaskIAMRoles to $true to enable task IAM roles
# Note that enabling IAM roles will make port 80 unavailable for tasks.
[bool]$EnableTaskIAMRoles = $false
if (${EnableTaskIAMRoles}) {
  $HostSetupScript = Invoke-WebRequest https://raw.githubusercontent.com/aws/amazon-
ecs-agent/master/misc/windows-deploy/hostsetup.ps1
  Invoke-Expression $($HostSetupScript.Content)
}
# Install the agent service
New-Service -Name "AmazonECS" `
        -BinaryPathName "$ecsExeDir\amazon-ecs-agent.exe -windows-service" `
        -DisplayName "Amazon ECS" `
        -Description "Amazon ECS service runs the Amazon ECS agent" `
        -DependsOn Docker `
        -StartupType Manual
sc.exe failure AmazonECS reset=300 actions=restart/5000/restart/30000/restart/60000
sc.exe failureflag AmazonECS 1
Start-Service AmazonECS
</powershell>
```

Utilisation d'un proxy HTTP pour les instances de conteneur Windows Amazon ECS

Vous pouvez configurer vos instances de conteneur Amazon ECS pour qu'elles utilisent un proxy HTTP pour l'agent de conteneur Amazon ECS et le démon Docker. C'est utile si vos instances de conteneur n'ont pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

Pour configurer votre instance de conteneur Windows Amazon ECS afin d'utiliser un proxy HTTP, définissez les variables suivantes au moment du lancement (avec les données EC2 utilisateur Amazon).

```
[Environment]::SetEnvironmentVariable("HTTP_PROXY",
"http://proxy.mydomain:port", "Machine")
```

Définissez `HTTP_PROXY` sur le nom d'hôte (ou l'adresse IP) et le numéro de port d'un proxy HTTP à utiliser pour que l'agent Amazon ECS puisse se connecter à Internet. Par exemple, vos instances de conteneur n'ont peut-être pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

```
[Environment]::SetEnvironmentVariable("NO_PROXY",
"169.254.169.254,169.254.170.2,\\.\pipe\docker_engine", "Machine")
```

Définissez cette `NO_PROXY` valeur `169.254.169.254,169.254.170.2,\\.\pipe\docker_engine` pour filtrer les métadonnées de l' EC2 instance, les rôles IAM pour les tâches et le trafic du daemon Docker provenant du proxy.

Example Script de données utilisateur proxy HTTP pour Windows

L'exemple de PowerShell script de données utilisateur ci-dessous configure l'agent de conteneur Amazon ECS et le daemon Docker pour utiliser un proxy HTTP que vous spécifiez. Vous pouvez également spécifier un cluster auprès duquel l'instance de conteneur s'enregistrera.

Pour utiliser ce script lors du lancement d'une instance de conteneur, suivez les étapes indiquées dans the section called “Lancement d'une instance de conteneur”. Copiez et collez simplement le PowerShell script ci-dessous dans le champ Données utilisateur (veillez à remplacer les exemples de valeurs en rouge par vos propres informations de proxy et de cluster).

> **Note**
>
> L'option `-EnableTaskIAMRole` est nécessaire pour activer les rôles IAM pour les tâches. Pour de plus amples informations, veuillez consulter Configuration supplémentaire de l'instance Amazon EC2 Windows.

```
<powershell>
Import-Module ECSTools

$proxy = "http://proxy.mydomain:port"
[Environment]::SetEnvironmentVariable("HTTP_PROXY", $proxy, "Machine")
[Environment]::SetEnvironmentVariable("NO_PROXY", "169.254.169.254,169.254.170.2,\\.
\pipe\docker_engine", "Machine")

Restart-Service Docker
Initialize-ECSAgent -Cluster MyCluster -EnableTaskIAMRole
</powershell>
```

Configuration des instances de conteneur Windows Amazon ECS pour recevoir des notifications relatives aux instances Spot

Amazon EC2 met fin, arrête ou met en veille prolongée votre instance Spot lorsque le prix Spot dépasse le prix maximum pour votre demande ou lorsque la capacité n'est plus disponible. Amazon EC2 fournit un avis d'interruption de l'instance Spot, qui donne à l'instance un avertissement de deux minutes avant son interruption. Si le drainage des instances Spot Amazon ECS est activé sur l'instance, ECS reçoit l'avis d'interruption d'instance Spot et bascule l'instance à l'état `DRAINING`.

> **Important**
>
> Amazon ECS surveille les avis d'interruption d'instance Spot qui comportent les actions d'instance `terminate` et `stop`. Si vous avez spécifié le comportement d'interruption d'instance `hibernate` lors de la demande de vos instances Spot ou de votre parc d'instances Spot, le drainage des instances Spot Amazon ECS n'est pas pris en charge pour ces instances.

Lorsqu'une instance de conteneur est définie sur `DRAINING`, Amazon ECS bloque la planification du placement des nouvelles tâches sur l'instance de conteneur. Les tâches de service ayant l'état

PENDING sur l'instance de conteneur faisant l'objet du drainage sont arrêtées immédiatement. S'il y a des instances de conteneur disponibles dans le cluster, des tâches de service de remplacement sont lancées dessus.

Vous pouvez activer le drainage des instances Spot lorsque vous lancez une instance. Vous devez définir le paramètre ECS_ENABLE_SPOT_INSTANCE_DRAINING avant de démarrer l'agent de conteneur. Remplacez *my-cluster* par le nom de votre cluster.

```
[Environment]::SetEnvironmentVariable("ECS_ENABLE_SPOT_INSTANCE_DRAINING", "true",
 "Machine")

# Initialize the agent
Initialize-ECSAgent -Cluster my-cluster
```

Pour de plus amples informations, veuillez consulter the section called “Lancement d'une instance de conteneur”.

Clusters Amazon ECS pour le type de lancement externe

Amazon ECS Anywhere fournit un support pour l'enregistrement d'une Instance externe, telle qu'un serveur sur site ou une machine virtuelle (VM), sur votre cluster Amazon ECS. Les instances externes sont optimisées pour exécuter des applications qui génèrent du trafic sortant ou des données de processus. Si votre application nécessite du trafic entrant, l'absence de prise en charge d'Elastic Load Balancing rend l'exécution de ces applications moins efficace. Amazon ECS a ajouté un nouveau type de lancement EXTERNAL que vous pouvez utiliser pour créer des services ou exécuter des tâches sur vos instances externes.

Systèmes d'exploitation et architectures système pris en charge

Les informations ci-dessous présentent la liste des systèmes d'exploitation et des architectures système pris en charge.

- Amazon Linux 2
- Amazon Linux 2023
- CentOS Stream 9
- RHEL 7, RHEL 8, RHEL 9 — Ni Docker ni les référentiels de packages ouverts de RHEL ne prennent en charge l'installation native de Docker sur RHEL. Vous devez vous assurer que Docker est installé avant d'exécuter le script d'installation décrit dans ce document.

- Fedora 32, Fedora 33, Fedora 40
- openSUSE Tumbleweed
- Ubuntu 18, Ubuntu 20, Ubuntu 22, Ubuntu 24
- Debian 10

> **Important**
>
> La prise en charge de Debian 9 Long Term (prise en charge LTS) a pris fin le 30 juin 2022 et n'est plus compatible avec Amazon ECS Anywhere.

- Debian 11
- Debian 12
- SUSE Enterprise Server 15
- Les architectures d'UC `x86_64` et `ARM64` sont prises en charge.
- Les versions de système d'exploitation Windows suivantes sont prises en charge :
  - Windows Server 2022
  - Windows Server 2019
  - Windows Server 2016
  - Windows Server 20H2

Considérations

Avant de démarrer l'utilisation des instances externes, tenez compte des considérations suivantes.

- Vous pouvez enregistrer une instance externe dans un cluster à la fois. Pour plus d'informations sur l'enregistrement d'une instance externe auprès d'un autre cluster, veuillez consulter Annulation de l'enregistrement d'une instance externe Amazon ECS.
- Vos instances externes ont besoin d'un rôle IAM qui leur permet de communiquer avec AWS APIs. Pour de plus amples informations, veuillez consulter Rôle IAM dans Amazon ECS Anywhere.
- Vos instances externes ne doivent pas avoir de chaîne d'informations d'identification d'instance préconfigurée définie localement, car cela interférera avec le script d'enregistrement.
- Pour envoyer des journaux de conteneur à CloudWatch Logs, assurez-vous de créer et de spécifier un rôle IAM d'exécution de tâche dans la définition de votre tâche.

- Lorsqu'une instance externe est enregistrée dans un cluster, la valeur `ecs.capability.external` est associée à l'instance. Cet attribut identifie l'instance en tant qu'instance externe. Vous pouvez ajouter des attributs personnalisés à vos instances externes pour les utiliser comme contrainte de placement de tâches. Pour de plus amples informations, veuillez consulter Attributs personnalisés.
- Vous pouvez ajouter des balises de ressources à votre instance externe. Pour de plus amples informations, veuillez consulter Instances de conteneurs externes.
- ECS Exec est pris en charge sur les instances externes. Pour de plus amples informations, veuillez consulter Surveillez les conteneurs Amazon ECS avec ECS Exec.
- Vous trouverez ci-dessous des considérations supplémentaires spécifiques aux réseaux avec vos instances externes. Pour de plus amples informations, veuillez consulter Réseaux .
  - La répartition de charge de service n'est pas prise en charge.
  - La découverte de service n'est pas prise en charge.
  - Les tâches qui s'exécutent sur des instances externes doivent utiliser les modes réseau `bridge`, `host` ou `none`. Le mode réseau `awsvpc` n'est pas pris en charge.
  - Il existe des domaines de service Amazon ECS dans chaque AWS région. Ces domaines de service doivent être autorisés à envoyer du trafic vers vos instances externes.
  - Le SSM Agent installé sur votre instance externe gère les informations d'identification IAM qui effectuent une rotation toutes les 30 minutes à l'aide d'une empreinte matérielle. Si votre instance externe perd la connexion à AWS, l'agent SSM actualise automatiquement les informations d'identification une fois la connexion rétablie. Pour de plus amples informations, veuillez consulter Validation de serveurs et de machines virtuelles sur site à l'aide d'une empreinte matérielle dans le Guide de l'utilisateur AWS Systems Manager .
- L'API `UpdateContainerAgent` n'est pas prise en charge. Pour obtenir des instructions sur la mise à jour du SSM Agent ou de l'agent Amazon ECS sur vos instances externes, veuillez consulter Mise à jour de l' AWS Systems Manager agent et de l'agent de conteneur Amazon ECS sur une instance externe.
- Les fournisseurs de capacité Amazon ECS ne sont pas pris en charge. Pour créer un service ou exécuter une tâche autonome sur vos instances externes, utilisez le type de lancement `EXTERNAL`.
- SELinux n'est pas pris en charge.
- L'utilisation de volumes Amazon EFS ou la spécification d'une `EFSVolumeConfiguration` n'est pas prise en charge.
- L'intégration à App Mesh n'est pas prise en charge.

- Si vous utilisez la console pour créer une définition de tâche d'instance externe, vous devez créer la définition de tâche avec l'éditeur JSON de la console.
- Lorsque vous exécutez ECS Anywhere sur Windows, vous devez utiliser votre propre licence Windows sur l'infrastructure sur site.
- Lorsque vous utilisez une AMI non optimisée pour Amazon ECS, exécutez les commandes suivantes sur l'instance de conteneur externe pour configurer les règles d'utilisation des rôles IAM pour les tâches. Pour de plus amples informations, veuillez consulter Configuration supplémentaire de l'instance externe.

```
$ sysctl -w net.ipv4.conf.all.route_localnet=1
$ iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
$ iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Réseaux

Les instances externes Amazon ECS sont optimisées pour exécuter des applications qui génèrent du trafic sortant ou des données de traitement. Si votre application nécessite un trafic entrant, tel qu'un service web, l'absence de prise en charge d'Elastic Load Balancing rend l'exécution de ces applications moins efficace, car il n'existe pas de prise en charge pour placer ces charges de travail derrière un équilibreur de charge.

Vous trouverez ci-dessous des considérations supplémentaires spécifiques aux réseaux avec vos instances externes.

- La répartition de charge de service n'est pas prise en charge.
- La découverte de service n'est pas prise en charge.
- Les tâches Linux qui s'exécutent sur des instances externes doivent utiliser les modes réseau `bridge`, `host` ou `none`. Le mode réseau `awsvpc` n'est pas pris en charge.

  Pour plus d'informations sur chaque mode réseau, consultez les options de mise en réseau des tâches Amazon ECS pour le type de EC2 lancement.
- Les tâches Windows qui s'exécutent sur des instances externes doivent utiliser le mode réseau `default`.
- Il existe des domaines de service Amazon ECS dans chaque région et vous devez être autorisé à envoyer du trafic vers vos instances externes.

- Le SSM Agent installé sur votre instance externe gère les informations d'identification IAM qui effectuent une rotation toutes les 30 minutes à l'aide d'une empreinte matérielle. Si votre instance externe perd la connexion à AWS, l'agent SSM actualise automatiquement les informations d'identification une fois la connexion rétablie. Pour plus d'informations, consultez Validation de serveurs et de machines virtuelles sur site à l'aide d'une empreinte matérielle dans le Guide de l'utilisateur AWS Systems Manager .

Les domaines suivants sont utilisés pour la communication entre le service Amazon ECS service et l'agent Amazon ECS installé sur votre instance externe. Assurez-vous que le trafic est autorisé et que la résolution DNS fonctionne. Pour chaque point de terminaison, *`region`* représente l'identifiant de AWS région d'une région prise en charge par Amazon ECS, par exemple `us-east-2` pour la région USA Est (Ohio). Les points de terminaison de toutes les régions que vous utilisez doivent être autorisés. Pour les points de terminaison `ecs-a` et `ecs-t`, vous devez inclure un astérisque (par exemple, `ecs-a-*`).

- `ecs-a-*.`*`region`*`.amazonaws.com` — Ce point de terminaison est utilisé lors de la gestion des tâches.
- `ecs-t-*.`*`region`*`.amazonaws.com` — Ce point de terminaison est utilisé pour gérer les métriques de tâche et de conteneur.
- `ecs.`*`region`*`.amazonaws.com` — Il s'agit du point de terminaison de service pour Amazon ECS.
- `ssm.`*`region`*`.amazonaws.com` — Il s'agit du point de terminaison du service pour AWS Systems Manager.
- `ec2messages.`*`region`*`.amazonaws.com`— Il s'agit du point de terminaison du service AWS Systems Manager utilisé pour communiquer entre l'agent Systems Manager et le service Systems Manager dans le cloud.
- `ssmmessages.`*`region`*`.amazonaws.com` — Il s'agit du point de terminaison requis pour créer et supprimer des canaux de session avec le service Session Manager dans le cloud.
- Si vos tâches nécessitent une communication avec d'autres AWS services, assurez-vous que ces points de terminaison de service sont autorisés. Parmi les applications, citons l'utilisation d'Amazon ECR pour extraire des images de conteneurs ou l'utilisation de CloudWatch for CloudWatch Logs. Pour plus d'informations, consultez Points de terminaison de service dans la Référence générale AWS .

Amazon FSx for Windows File Server avec ECS Anywhere

Pour utiliser les instances externes Amazon FSx for Windows File Server d'Amazon ECS, vous devez établir une connexion entre votre centre de données sur site et le AWS Cloud. Pour plus d'informations sur la connexion de votre réseau à votre VPC, consultez Options de connectivité du cloud privé virtuel Amazon.

gMSA avec ECS Anywhere

Les cas d'utilisation suivants sont pris en charge pour ECS Anywhere.

- L'Active Directory se trouve dans le AWS Cloud - Pour cette configuration, vous créez une connexion entre votre réseau local et l' AWS Cloud utilisation d'une AWS Direct Connect connexion. Pour en savoir plus comment créer une connexion, consultez Options de connectivité du cloud privé virtuel Amazon. Vous pouvez créer un Active Directory dans AWS Cloud. Pour plus d'informations sur la façon de démarrer AWS Directory Service, consultez la section Configuration AWS Directory Service du Guide d'AWS Directory Service administration. Vous pouvez ensuite joindre vos instances externes au domaine à l'aide de la AWS Direct Connect connexion. Pour en savoir plus sur l'utilisation de gMSA avec Amazon ECS, consultez the section called “Apprenez à utiliser g MSAs pour les conteneurs EC2 Windows”.
- Active Directory se trouve dans le centre de données sur site. - Pour cette configuration, vous joignez vos instances externes à Active Directory sur site. Vous utilisez ensuite les informations d'identification disponibles localement lorsque vous exécutez les tâches Amazon ECS.

Création d'un cluster Amazon ECS pour le type de lancement externe

Vous créez un cluster pour définir l'infrastructure sur laquelle s'exécutent vos tâches et services.

Avant de commencer, veillez achever les étapes de Configurer l'utilisation d'Amazon ECS et affectez l’autorisation IAM appropriée. Pour de plus amples informations, veuillez consulter the section called “Exemples de clusters Amazon ECS”. La console Amazon ECS fournit un moyen simple de créer les ressources nécessaires à un cluster Amazon ECS en créant une AWS CloudFormation pile.

Pour que le processus de création de cluster soit aussi simple que possible, la console dispose de sélections par défaut pour de nombreux choix que nous décrivons ci-dessous. La plupart des sections de la console comportent des volets d'aide qui fournissent un contexte supplémentaire.

Vous pouvez modifier les options suivantes :

- Ajoutez un espace de noms au cluster.

  Un espace de noms permet aux services que vous créez dans le cluster de se connecter aux autres services de l'espace de noms sans configuration supplémentaire. Pour de plus amples informations, veuillez consulter Interconnectez les services Amazon ECS.

- Configuration du cluster pour les instances externes
- Activez Container Insights avec une observabilité améliorée, ou Container Insights.

  CloudWatch Container Insights collecte, agrège et résume les métriques et les journaux de vos applications conteneurisées et de vos microservices. Conteneur Insights fournit également des informations de diagnostic (par exemple sur les échecs de redémarrage des conteneurs) que vous pouvez utiliser pour isoler les problèmes et les résoudre rapidement. Pour de plus amples informations, veuillez consulter the section called “Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée”.

  Le 2 décembre 2024, AWS a publié Container Insights avec une observabilité améliorée pour Amazon ECS. Cette version prend en charge une observabilité améliorée pour les clusters Amazon ECS utilisant les types de lancement Amazon EC2 et Fargate. Une fois que vous avez configuré Container Insights avec une observabilité améliorée sur Amazon ECS, Container Insights collecte automatiquement des données télémétriques détaillées sur l'infrastructure, du niveau du cluster au niveau du conteneur dans votre environnement, et affiche vos données dans des tableaux de bord qui vous présentent diverses mesures et dimensions. Vous pouvez ensuite utiliser ces out-of-the-box tableaux de bord sur la console Container Insights pour mieux comprendre l'état et les performances de vos conteneurs, et pour atténuer les problèmes plus rapidement en identifiant les anomalies.

  Nous vous recommandons d'utiliser Container Insights avec une observabilité améliorée plutôt que Container Insights, car cela fournit une visibilité détaillée de votre environnement de conteneurs, réduisant ainsi le délai moyen de résolution.

- Attribuez une AWS KMS clé à votre espace de stockage géré. Pour plus d'informations sur la création d'une clé, voir Création d'une clé KMS dans le Guide de AWS Key Management Service l'utilisateur.
- Ajoutez des balises pour vous aider à identifier vos clusters.

Pour créer un nouveau cluster (console Amazon ECS)

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.

2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Configuration de cluster, configurez les éléments suivants :
   - Pour Nom du cluster, saisissez un nom unique.

     Le nom peut contenir jusqu'à 255 lettres (minuscules et majuscules), des chiffres et des traits d'union.
   - (Facultatif) Pour que l'espace de noms utilisé pour Service Connect soit différent du nom du cluster, saisissez un nom unique dans Espace de nom.
6. (Facultatif) Utilisez Container Insights, développez Monitoring, puis choisissez l'une des options suivantes :
   - Pour utiliser le Container Insights recommandé avec une observabilité améliorée, choisissez Container Insights avec une observabilité améliorée.
   - Pour utiliser Container Insights, choisissez Container Insights.
7. (Facultatif) Chiffrez les données sur le stockage géré. Sous Chiffrement, pour Stockage géré, entrez l'ARN de la AWS KMS clé que vous souhaitez utiliser pour chiffrer les données du stockage géré.
8. (Facultatif) Pour vous aider à identifier votre cluster, développez Tags (balises), puis configurez vos balises.

   [Add a tag] Choisissez Add tag (Ajouter une balise) et procédez comme suit :
   - Pour Key (Clé), saisissez le nom de la clé.
   - Pour Valeur, saisissez la valeur de clé.
9. Choisissez Créer.

Étapes suivantes

Vous devez enregistrer les instances auprès du cluster. Pour de plus amples informations, veuillez consulter Enregistrement d'une instance externe dans un cluster Amazon ECS.

Créez une définition de tâche pour le type de lancement externe. Pour de plus amples informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Exécutez vos applications en tant que tâches autonomes ou dans le cadre d'un service. Pour plus d'informations, consultez les ressources suivantes :

- Exécution d'une application en tant que tâche Amazon ECS
- Création d'un déploiement de mises à jour continues Amazon ECS

Enregistrement d'une instance externe dans un cluster Amazon ECS

Pour chaque instance externe que vous enregistrez auprès d'un cluster Amazon ECS, le SSM Agent, l'agent de conteneur Amazon ECS et Docker doivent être installés. Pour enregistrer l'instance externe dans un cluster Amazon ECS, elle doit d'abord être enregistrée en tant qu'instance AWS Systems Manager gérée. Vous pouvez créer le script d'installation en quelques clics sur la console Amazon ECS. Le script d'installation inclut une clé d'activation Systems Manager et des commandes pour installer chacun des agents requis et Docker. Le script d'installation doit être exécuté sur votre serveur local ou votre machine virtuelle pour terminer la procédure d'installation et d'enregistrement.

> **Note**
>
> Avant d'enregistrer votre instance externe Linux auprès du cluster, créez le fichier `/etc/ecs/ecs.config` sur votre instance externe et ajoutez les paramètres de configuration de l'agent de conteneur souhaités. Vous ne pouvez pas le faire après l'enregistrement de l'instance externe dans un cluster. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

AWS Management Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Dans la page Clusters, sélectionnez un cluster dans lequel enregistrer votre instance externe.
5. Sur la *name* page Cluster :, choisissez l'onglet Infrastructure.
6. Sur la page Register external instances (Enregistrer des instances externes), effectuez les étapes suivantes.

   a. Pour Activation key duration (in days) [Durée de la clé d'activation (en jours)], saisissez le nombre de jours pendant lesquels la clé d'activation reste active. Une fois que le

nombre de jours que vous avez saisis est écoulé, la clé ne fonctionne plus lors de l'enregistrement d'une instance externe.

b. Pour Number of instances (Nombre d'instances), saisissez le nombre d'instances externes que vous souhaitez enregistrer dans votre cluster avec la clé d'activation.

c. Pour Instance role (Rôle d'instance), choisissez le rôle IAM à associer à vos instances externes. Si aucun rôle n'a déjà été créé, choisissez Create new role (Créer un rôle) Pour qu'Amazon ECS crée un rôle en votre nom. Pour plus d'informations sur les autorisations IAM requises pour vos instances externes, consultez Rôle IAM dans Amazon ECS Anywhere.

d. Copiez la commande d'inscription. Cette commande doit être exécutée sur chaque instance externe que vous souhaitez enregistrer dans le cluster.

> **Important**
>
> La partie bash du script doit être exécutée en tant que root. Si la commande n'est pas exécutée en tant que racine, une erreur est renvoyée.

e. Choisissez Close (Fermer).

AWS CLI for Linux operating systems

1. Créez une paire d'activations de Systems Manager. Elle est utilisée pour l'activation de l'instance gérée par Systems Manager. La sortie inclut un `ActivationId` et un `ActivationCode`. Vous les utiliserez dans une étape ultérieure. Veillez à spécifier le rôle IAM ECS Anywhere que vous avez créé. Pour de plus amples informations, veuillez consulter Rôle IAM dans Amazon ECS Anywhere.

```
aws ssm create-activation --iam-role ecsAnywhereRole | tee ssm-activation.json
```

2. Sur votre serveur local ou votre machine virtuelle (VM), téléchargez le script d'installation.

```
curl --proto "https" -o "/tmp/ecs-anywhere-install.sh" "https://amazon-ecs-
agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh"
```

3. (Facultatif) Sur votre serveur sur site ou votre machine virtuelle (VM), procédez comme suit pour vérifier le script d'installation à l'aide du fichier de signature du script.

a. Téléchargez et installez GnuPG. Pour plus d'informations GNUpg, consultez le site Web de GnuPG. Pour les systèmes Linux, installez gpg à l'aide du gestionnaire de package de votre version de Linux.

b. Extrayez la clé publique PGP Amazon ECS.

```
gpg --keyserver hkp://keys.gnupg.net:80 --recv BCE9D9A42D51784F
```

c. Téléchargez la signature du script d'installation. La signature est une signature PGP détachées ASCII, stockée dans un fichier portant l'extension `.asc`.

```
curl --proto "https" -o "/tmp/ecs-anywhere-install.sh.asc" "https://amazon-
ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh.asc"
```

d. Vérifiez le fichier de script d'installation à l'aide de la clé.

```
gpg --verify /tmp/ecs-anywhere-install.sh.asc /tmp/ecs-anywhere-install.sh
```

La sortie attendue est la suivante :

```
gpg: Signature made Tue 25 May 2021 07:16:29 PM UTC
gpg:                using RSA key 50DECCC4710E61AF
gpg: Good signature from "Amazon ECS <ecs-security@amazon.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the
 owner.
Primary key fingerprint: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: D64B B6F9 0CF3 77E9 B5FB  346F 50DE CCC4 710E 61AF
```

4. Sur votre serveur sur site ou votre machine virtuelle (VM), exécutez le script d'installation. Spécifiez le nom du cluster, la région, l'ID d'activation de Systems Manager et le code d'activation à partir de la première étape.

```
sudo bash /tmp/ecs-anywhere-install.sh \
    --region $REGION \
    --cluster $CLUSTER_NAME \
    --activation-id $ACTIVATION_ID \
    --activation-code $ACTIVATION_CODE
```

Pour un serveur local ou une machine virtuelle (VM) sur lequel le pilote NVIDIA est installé pour les charges de travail GPU, vous devez ajouter l'indicateur `--enable-gpu` du script d'installation. Lorsque cet indicateur est spécifié, le script d'installation vérifie que le pilote NVIDIA est en cours d'exécution, puis ajoute les variables de configuration requises pour exécuter vos tâches Amazon ECS. Pour plus d'informations sur l'exécution de charges de travail GPU et la spécification des exigences GPU dans une définition de tâche, consultez Spécification GPUs dans une définition de tâche Amazon ECS.

```
sudo bash /tmp/ecs-anywhere-install.sh \
    --region $REGION \
    --cluster $CLUSTER_NAME \
    --activation-id $ACTIVATION_ID \
    --activation-code $ACTIVATION_CODE \
    --enable-gpu
```

Pour enregistrer une instance externe existante auprès d'un cluster différent, procédez comme suit.

Pour enregistrer une instance externe existante auprès d'un cluster différent

1. Arrêtez l'agent de conteneur Amazon ECS.

```
sudo systemctl stop ecs.service
```

2. Modifiez le fichier `/etc/ecs/ecs.config` et sur la ligne `ECS_CLUSTER`, assurez-vous que le nom du cluster correspond au nom du cluster auprès duquel vous souhaitez enregistrer l'instance externe.
3. Supprimez les données existantes de l'agent Amazon ECS.

```
sudo rm /var/lib/ecs/data/agent.db
```

4. Démarrez l'agent de conteneur Amazon ECS.

```
sudo systemctl start ecs.service
```

AWS CLI for Windows operating systems

1. Créez une paire d'activations de Systems Manager. Elle est utilisée pour l'activation de l'instance gérée par Systems Manager. La sortie inclut un `ActivationId` et un `ActivationCode`. Vous les utiliserez dans une étape ultérieure. Veillez à spécifier le rôle IAM ECS Anywhere que vous avez créé. Pour de plus amples informations, veuillez consulter Rôle IAM dans Amazon ECS Anywhere.

```
aws ssm create-activation --iam-role ecsAnywhereRole | tee ssm-activation.json
```

2. Sur votre serveur local ou votre machine virtuelle (VM), téléchargez le script d'installation.

```
Invoke-RestMethod -URI "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-
install.ps1" -OutFile “ecs-anywhere-install.ps1”
```

3. (Facultatif) Le script PowerShell est signé par Amazon et, par conséquent, Windows effectue automatiquement la validation du certificat sur ce dernier. Aucune validation manuelle n'est requise.

   Pour vérifier manuellement le certificat, effectuez un clic droit sur le fichier, accédez aux propriétés et utilisez l'onglet Signatures numériques pour obtenir plus de détails.

   Cette option n'est disponible que lorsque l'hôte possède le certificat dans le magasin de certificats.

   La vérification doit renvoyer des informations semblables à ce qui suit :

```
# Verification (PowerShell)
Get-AuthenticodeSignature -FilePath .\ecs-anywhere-install.ps1

SignerCertificate                         Status      Path
-----------------                         ------      ----
EXAMPLECERTIFICATE                        Valid       ecs-anywhere-install.ps1

...

Subject              : CN="Amazon Web Services, Inc.",...

----
```

4. Sur votre serveur sur site ou votre machine virtuelle (VM), exécutez le script d'installation. Spécifiez le nom du cluster, la région, l'ID d'activation de Systems Manager et le code d'activation à partir de la première étape.

```
.\ecs-anywhere-install.ps1 -Region $Region -Cluster $Cluster -
ActivationID $ActivationID -ActivationCode $ActivationCode
```

5. Vérifiez que l'agent de conteneur Amazon ECS est en cours d'exécution.

```
Get-Service AmazonECS

Status   Name               DisplayName
------   ----               -----------
Running  AmazonECS          Amazon ECS
```

Pour enregistrer une instance externe existante auprès d'un cluster différent, procédez comme suit.

Pour enregistrer une instance externe existante auprès d'un cluster différent

1. Arrêtez l'agent de conteneur Amazon ECS.

```
Stop-Service AmazonECS
```

2. Modifier le paramètre `ECS_CLUSTER` afin que le nom du cluster corresponde au nom du cluster auprès duquel vous souhaitez enregistrer l'instance externe.

```
[Environment]::SetEnvironmentVariable("ECS_CLUSTER", $ECSCluster,
 [System.EnvironmentVariableTarget]::Machine)
```

3. Supprimez les données existantes de l'agent Amazon ECS.

```
Remove-Item -Recurse -Force $env:ProgramData\Amazon\ECS\data\*
```

4. Démarrez l'agent de conteneur Amazon ECS.

```
Start-Service AmazonECS
```

Il AWS CLI peut être utilisé pour créer une activation de Systems Manager avant d'exécuter le script d'installation afin de terminer le processus d'enregistrement de l'instance externe.

Annulation de l'enregistrement d'une instance externe Amazon ECS

Nous vous recommandons de désenregistrer l'instance à la fois auprès d'Amazon ECS et une AWS Systems Manager fois que vous en aurez terminé avec l'instance. Après l'annulation de l'enregistrement, l'instance externe n'est plus en mesure d'accepter de nouvelles tâches.

Si des tâches sont en cours d'exécution sur l'instance de conteneur lorsque vous annulez l'enregistrement, ces tâches restent en cours d'exécution jusqu'à ce que vous les arrêtiez d'une autre manière. Toutefois, ces tâches ne sont plus surveillées ou prise en compte par Amazon ECS. Si ces tâches sur votre instance externe font partie d'un Amazon ECS service, le planificateur de service commence une autre copie de cette tâche sur une autre instance de conteneur, si possible.

Après avoir désenregistré l'instance, nettoyez les AWS ressources restantes sur l'instance. Vous pouvez ensuite l'enregistrer dans un nouveau cluster.

Procédure

AWS Management Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, choisissez la région dans laquelle votre instance externe est inscrite.
3. Dans le panneau de navigation, choisissez Clusters, puis sélectionnez le cluster qui héberge l'instance externe.
4. Sur la *name* page Cluster :, choisissez l'onglet Infrastructure.
5. Sous Constainer instances (Instances de conteneur), sélectionnez l'ID de l'instance externe pour annuler l'enregistrement. Vous êtes redirigé vers la page de détails de l'instance de conteneur.
6. Sur la *id* page Container Instance :, choisissez Désenregistrer.
7. Passez en revue le message d'annulation d'enregistrement. Sélectionnez Deregister from AWS Systems Manager(Annuler l'enregistrement AWS Systems Manager) pour également annuler l'enregistrement de l'instance externe en tant qu'instance gérée par Systems Manager. Choisissez Deregister (Annuler l'enregistrement).

> **Note**
>
> Vous pouvez annuler l'enregistrement de l'instance externe en tant qu'instance gérée par Systems Manager dans la console Systems Manager. Pour obtenir des instructions, consultez la section Désenregistrement des nœuds gérés dans un environnement hybride et multicloud dans le guide de l'utilisateur.AWS Systems Manager

8. Après avoir désenregistré l'instance, nettoyez les AWS ressources sur votre serveur local ou sur votre machine virtuelle.

Système d'exploitation	Étapes	

Linux	a. Arrêtez l'agent de conteneur Amazon ECS et les services SSM Agent sur l'instance.
	`sudo systemctl stop ecs amazon-ssm-agent`
	b. Supprimez les packages Amazon ECS et Systems Manager.
	Pour CentOS 7, CentOS 8 et RHEL 7
	`sudo yum remove -y amazon-ecs-init amazon-ssm-agent`
	Pour SUSE Enterprise Server 15	

Système d’exploitation	Étapes

	`sudo zypper remove -y amazon-ecs-init amazon-ssm-agent`
	
	Pour Debian et Ubuntu
	
	`sudo apt remove -y amazon-ecs-init amazon-ssm-agent`
	
	c. Supprimez les répertoires restants.
	
	`sudo rm -rf /var/lib/ecs /etc/ecs /var/lib/amazon/ssm /var/log/ecs /var/log/amazon/ssm`

Windows	a. Arrêtez l'agent de conteneur Amazon ECS et les services SSM Agent sur l'instance.
	
	`Stop-Service AmazonECS`
	
	`Stop-Service AmazonSSMAgent`
	
	b. Supprimez le package Amazon ECS.
	
	`.\ecs-anywhere-install.ps1 -Uninstall`

AWS CLI

1. Vous avez besoin de l'ID d'instance et de l'ARN de l'instance de conteneur pour annuler l'enregistrement de l'instance de conteneur. Si vous n'avez pas ces valeurs, exécutez les commandes suivantes.

   Exécutez la commande suivante pour obtenir l'ID d'instance.

   Vous utilisez l'ID d'instance (`instanceID`) pour obtenir l'ARN de l'instance de conteneur (`containerInstanceARN`).

   ```
   instanceId=$(aws ssm describe-instance-information --region "{{ region }}" |
    jq ".InstanceInformationList[] |select(.IPAddress==\"{{ IPv4 Address }}\")
    | .InstanceId" | tr -d'"'
   ```

   Exécutez les commandes suivantes.

   Vous utilisez l'`containerInstanceArn` comme paramètre dans la commande pour annuler l'enregistrement de l'instance (`deregister-container-instance`).

   ```
   instances=$(aws ecs list-container-instances --cluster "{{ cluster }}" --region
    "{{ region }}" | jq -c '.containerInstanceArns')
   containerInstanceArn=$(aws ecs describe-container-instances --cluster
    "{{ cluster }}" --region "{{ region }}" --container-instances $instances
    | jq ".containerInstances[] | select(.ec2InstanceId==\"{{ instanceId }}\")
    | .containerInstanceArn" | tr -d '"')
   ```

2. Exécutez la commande suivante pour purger l'instance.

   ```
   aws ecs update-container-instances-state --cluster "{{ cluster }}" --region
    "{{ region }}" --container-instances "{{ containerInstanceArn }}" --status
    DRAINING
   ```

3. Une fois le drainage de l'instance de conteneur terminé, exécutez la commande suivante pour annuler son enregistrement.

   ```
   aws ecs deregister-container-instance --cluster "{{ cluster }}" --region
    "{{ region }}" --container-instance "{{ containerInstanceArn }}"
   ```

4. Exécutez la commande suivante pour supprimer l'instance de conteneur de SSM.

```
aws ssm deregister-managed-instance --region "{{ region }}" --instance-id
 "{{ instanceId }}"
```

5. Après avoir désenregistré l'instance, nettoyez les AWS ressources sur votre serveur local ou sur votre machine virtuelle.

Système d'exploitation	Étapes	

Linux	a. Arrêtez l'agent de conteneur Amazon ECS et les services SSM Agent sur l'instance.
	`sudo systemctl stop ecs amazon-ssm-agent`
	b. Supprimez les packages Amazon ECS et Systems Manager.
	`sudo (yum/apt/zypper) remove amazon-ecs-init amazon-ssm-agent`
	c. Supprimez les répertoires restants.
	`sudo rm -rf /var/lib/ecs /etc/ecs /var/lib/amazon/ssm /var/log/ecs /var/log/amazon/ssm`	

Windows	a. Arrêtez l'agent de conteneur Amazon ECS et les services SSM Agent sur l'instance.	

Système d'exploitation	Étapes	

	`Stop-Service AmazonECS`
	
	`Stop-Service AmazonSSMAgent`
	
	b. Supprimez le package Amazon ECS.
	
	`.\ecs-anywhere-install.ps1 -Uninstall`	

Mise à jour de l' AWS Systems Manager agent et de l'agent de conteneur Amazon ECS sur une instance externe

Votre serveur ou machine virtuelle sur site doit exécuter à la fois l' AWS Systems Manager agent (agent SSM) et l'agent de conteneur Amazon ECS lors de l'exécution de charges de travail Amazon ECS. AWS publie de nouvelles versions de ces agents lorsque des fonctionnalités sont ajoutées ou mises à jour. Si vos instances externes utilisent une version antérieure de l'un ou l'autre des agents, vous pouvez les mettre à jour à l'aide des procédures suivantes.

Mise à jour de SSM Agent sur une instance externe

AWS Systems Manager vous recommande d'automatiser le processus de mise à jour de l'agent SSM sur vos instances. Ils fournissent plusieurs méthodes pour automatiser les mises à jour. Pour de plus amples informations, veuillez consulter Automatisation des mises à jour sur SSM Agent dans le Guide de l'utilisateur AWS Systems Manager .

Mise à jour de l'agent Amazon ECS sur une instance externe

Sur vos instances externes, l'agent de conteneur Amazon ECS est mis à jour en mettant à niveau le package `ecs-init`. La mise à jour de l'agent Amazon ECS n'interrompt pas les tâches ou les services en cours d'exécution. Amazon ECS fournit le package `ecs-init` et le fichier SIGNATURE dans un compartiment Amazon S3 dans chaque région. En commençant par `ecs-init` version

`1.52.1-1`, Amazon ECS fournit des packages `ecs-init` distincts à utiliser en fonction du système d'exploitation et de l'architecture système que votre instance externe utilise.

Utilisez le tableau suivant pour déterminer le package `ecs-init` que vous devez télécharger en fonction du système d'exploitation et de l'architecture système que votre instance externe utilise.

> **Note**
>
> Vous pouvez déterminer le système d'exploitation et l'architecture système que votre instance externe utilise à l'aide des commandes suivantes.
>
> ```
> cat /etc/os-release
> uname -m
> ```

Systèmes d'exploitation (architecture)	package ecs-init
CentOS 7 (x86_64) CentOS 8 (x86_64) CentOS Stream 9 (x86_64) SUSE Enterprise Server 15 (x86_64) RHEL 7 (x86_64) RHEL 8 (x86_64)	`amazon-ecs-init-latest.x86_64.rpm`
CentOS 7 (aarch64) CentOS 8 (aarch64) CentOS Stream 9 (aarch64) RHEL 7 (aarch64)	`amazon-ecs-init-latest.aarch64.rpm`
Debian 9 (x86_64) Debian 10 (x86_64)	`amazon-ecs-init-latest.amd64.deb`

Systèmes d'exploitation (architecture)	package ecs-init
Debian 11 (x86_64) Debian 12 (x86_64) Ubuntu 18 (x86_64) Ubuntu 20 (x86_64) Ubuntu 22 (x86_64) Ubuntu 24 (x86_64)	
Debian 9 (aarch64) Debian 10 (aarch64) Debian 11 (aarch64) Debian 12 (aarch64) Ubuntu 18 (aarch64) Ubuntu 20 (aarch64) Ubuntu 22 (aarch64) Ubuntu 24 (aarch64)	`amazon-ecs-init-latest.arm64.deb`

Procédez comme suit pour mettre à jour l'agent Amazon ECS.

Pour mettre à jour l'agent Amazon ECS

1. Confirmez la version de l'agent Amazon ECS que vous exécutez.

```
curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
```

2. Télécharger le package `ecs-init` pour votre système d'exploitation et votre architecture système. Amazon ECS fournit le fichier de package `ecs-init` dans un compartiment Amazon S3 dans chaque région. Assurez-vous de remplacer l'*<region>*identifiant dans la

commande par le nom de la région (par exemple`us-west-2`) dont vous êtes le plus proche géographiquement.

amazon-ecs-init-latest.x86_64 .rpm

```
curl -o amazon-ecs-init.rpm https://s3.<region>.amazonaws.com/amazon-ecs-
agent-<region>/amazon-ecs-init-latest.x86_64.rpm
```

amazon-ecs-init-latest.aarch64 tr/min

```
curl -o amazon-ecs-init.rpm https://s3.<region>.amazonaws.com/amazon-ecs-
agent-<region>/amazon-ecs-init-latest.aarch64.rpm
```

amazon-ecs-init-latest.amd64.deb

```
curl -o amazon-ecs-init.deb https://s3.<region>.amazonaws.com/amazon-ecs-
agent-<region>/amazon-ecs-init-latest.amd64.deb
```

amazon-ecs-init-latest.arm64.deb

```
curl -o amazon-ecs-init.deb https://s3.<region>.amazonaws.com/amazon-ecs-
agent-<region>/amazon-ecs-init-latest.arm64.deb
```

3. (Facultatif) Vérification de la validité du fichier de package `ecs-init` à l'aide de la signature PGP.

   a. Téléchargez et installez GnuPG. Pour plus d'informations GNUpg, consultez le site Web de GnuPG. Pour les systèmes Linux, installez gpg à l'aide du gestionnaire de package de votre version de Linux.

   b. Extrayez la clé publique PGP Amazon ECS.

   ```
   gpg --keyserver hkp://keys.gnupg.net:80 --recv BCE9D9A42D51784F
   ```

   c. Téléchargez la signature de package `ecs-init`. La signature est une signature PGP détachées ASCII, stockée dans un fichier portant l'extension `.asc`. Amazon ECS fournit le fichier SIGNATURE dans un compartiment Amazon S3 dans chaque région. Assurez-vous de remplacer l'`<region>`identifiant dans la commande par le nom de la région (par exemple`us-west-2`) dont vous êtes le plus proche géographiquement.

amazon-ecs-init-latest.x86_64 .rpm

```
curl -o amazon-ecs-init.rpm.asc https://s3.<region>.amazonaws.com/amazon-ecs-
agent-<region>/amazon-ecs-init-latest.x86_64.rpm.asc
```

amazon-ecs-init-latest.aarch64 tr/min

```
curl -o amazon-ecs-init.rpm.asc https://s3.<region>.amazonaws.com/amazon-ecs-
agent-<region>/amazon-ecs-init-latest.aarch64.rpm.asc
```

amazon-ecs-init-latest.amd64.deb

```
curl -o amazon-ecs-init.deb.asc https://s3.<region>.amazonaws.com/amazon-ecs-
agent-<region>/amazon-ecs-init-latest.amd64.deb.asc
```

amazon-ecs-init-latest.arm64.deb

```
curl -o amazon-ecs-init.deb.asc https://s3.<region>.amazonaws.com/amazon-ecs-
agent-<region>/amazon-ecs-init-latest.arm64.deb.asc
```

d. Vérifiez le fichier de package `ecs-init` à l'aide de la clé.

Pour les packages **`rpm`**

```
gpg --verify amazon-ecs-init.rpm.asc ./amazon-ecs-init.rpm
```

Pour les packages **`deb`**

```
gpg --verify amazon-ecs-init.deb.asc ./amazon-ecs-init.deb
```

La sortie attendue est la suivante :

```
gpg: Signature made Fri 14 May 2021 09:31:36 PM UTC
gpg:                using RSA key 50DECCC4710E61AF
gpg: Good signature from "Amazon ECS <ecs-security@amazon.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
```

```
      Subkey fingerprint: D64B B6F9 0CF3 77E9 B5FB  346F 50DE CCC4 710E 61AF
```

4. Installez le package `ecs-init`.

   Pour le package **`rpm`** sur CentOS 7, CentOS 8 et RHEL 7

   ```
   sudo yum install -y ./amazon-ecs-init.rpm
   ```

   Pour le package **`rpm`** sur SUSE Enterprise Server 15

   ```
   sudo zypper install -y --allow-unsigned-rpm ./amazon-ecs-init.rpm
   ```

   Pour le package **`deb`**

   ```
   sudo dpkg -i ./amazon-ecs-init.deb
   ```

5. Redémarrez le service `ecs`.

   ```
   sudo systemctl restart ecs
   ```

6. Vérifiez que la version d'agent Amazon ECS a été mise à jour.

   ```
   curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
   ```

Mise à jour d'un cluster Amazon ECS

Vous pouvez modifier les propriétés de cluster suivantes :

- Définissez un fournisseur de capacité par défaut.

  Chaque cluster dispose d'un ou plusieurs fournisseurs de capacité et éventuellement d'une stratégie de fournisseur de capacité facultative. La stratégie de fournisseur de capacité détermine la façon dont les tâches sont réparties entre les fournisseurs de capacité du cluster. Lorsque vous exécutez une tâche autonome ou que vous créez un service, vous pouvez utiliser soit la stratégie de fournisseur de capacité par défaut du cluster, soit une stratégie de fournisseur de capacité qui remplace celle par défaut.

- Activez Container Insights.

CloudWatch Container Insights collecte, agrège et résume les métriques et les journaux de vos applications conteneurisées et de vos microservices. Conteneur Insights fournit également des informations de diagnostic (par exemple sur les échecs de redémarrage des conteneurs) que vous pouvez utiliser pour isoler les problèmes et les résoudre rapidement. Pour de plus amples informations, veuillez consulter the section called “Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée”.

- Ajoutez des étiquettes pour vous aider à identifier vos clusters.

Procédure

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur la *name* page Cluster :, choisissez Mettre à jour le cluster.
5. Pour définir le fournisseur de capacité par défaut, sous Stratégie de fournisseur de capacité par défaut, sélectionnez Ajouter plus.

   a. Pour Fournisseur de capacité, choisissez le fournisseur de capacité.

   b. (Facultatif) Pour Base, saisissez le nombre minimal de tâches exécutées sur le fournisseur de capacité.

      Vous ne pouvez définir une valeur de Base que pour un seul fournisseur de capacité.

   c. (Facultatif) Pour Poids, saisissez le pourcentage relatif du nombre total de tâches lancées qui utilisent le fournisseur de capacité spécifié.

   d. (Facultatif) Répétez les étapes pour tous les fournisseurs de capacité supplémentaires.
6. Pour activer ou désactiver Container Insights, développez Surveillance, puis activez Utiliser Container Insights.
7. Pour vous aider à identifier votre cluster, développez Balises, puis configurez vos balises.

   [Add a tag] Choisissez Add tag (Ajouter une balise) et procédez comme suit :

   - Pour Key (Clé), saisissez le nom de la clé.
   - Pour Value (Valeur), saisissez la valeur de clé.

   [Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de la balise.

8. Choisissez Mettre à jour.

Suppression d'un cluster Amazon ECS

Si vous avez terminé d'utiliser un cluster, vous pouvez le supprimer. Une fois que vous avez supprimé le cluster, il passe à l'état `INACTIVE`. Les clusters ayant un état `INACTIVE` peuvent rester détectables dans votre compte pendant un certain temps. Cependant, cela est susceptible de changer à terme. Il est donc important de ne pas compter sur la persistance des clusters `INACTIVE`.

Avant de supprimer un cluster, vous devez effectuer les opérations suivantes :

- Supprimez tous les services du cluster. Pour de plus amples informations, veuillez consulter the section called “Suppression d'un service”.
- Arrêtez toutes les tâches en cours d'exécution. Pour de plus amples informations, veuillez consulter the section called “Arrêt d'une tâche”.
- Annulez l'enregistrement de toutes les instances de conteneur enregistrées dans le cluster. Pour de plus amples informations, veuillez consulter the section called “Annulation de l'enregistrement d'une instance de conteneur”.
- Supprimez l'espace de noms. Pour plus d'informations, consultez Suppression des espaces de noms dans le Guide du développeur AWS Cloud Map .

Procédure

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters sélectionnez le cluster à supprimer.
5. Dans le coin supérieur droit de la page, choisissez Delete Cluster (Supprimer le cluster).

   Un message s'affiche lorsque vous n'avez pas supprimé toutes les ressources associées au cluster.
6. Dans le champ de confirmation, saisissez Supprimer ***`cluster name`***.

Annulation de l'enregistrement d'une instance de conteneur Amazon ECS

> **Important**
>
> Cette rubrique concerne EC2 uniquement les instances de conteneur créées sur Amazon. Pour plus d'informations sur l'annulation de l'enregistrement des instances externes, veuillez consulterAnnulation de l'enregistrement d'une instance externe Amazon ECS.

Lorsque vous avez terminé d'utiliser une instance de conteneur EC2 sauvegardée par Amazon, vous devez la désenregistrer de votre cluster. Après l'annulation de l'enregistrement, l'instance de conteneur n'est plus en mesure d'accepter de nouvelles tâches.

Si des tâches sont en cours d'exécution sur l'instance du conteneur lorsque vous annulez l'enregistrement, ces tâches restent en cours d'exécution jusqu'à ce que vous résiliez l'instance ou les tâches d'une autre manière. Toutefois, ces tâches sont orphelines, ce qui signifie qu'elles ne sont plus surveillées ou prise en compte par Amazon ECS. Si une tâche orpheline sur votre instance de conteneur fait partie d'un service Amazon ECS service, le planificateur de service commence une autre copie de cette tâche sur une autre instance de conteneur, si possible. L'enregistrement de tout conteneur des tâches de service orphelines enregistré avec un groupe cible Application Load Balancer est annulé. Le drainage de la connexion commence en fonction des paramètres de l'équilibreur de charge ou du groupe cible. Si une tâche orpheline utilise le mode réseau `awsvpc`, leurs interfaces réseau Elastic sont supprimées.

Si vous prévoyez d'utiliser l'instance de conteneur à d'autres fins après l'annulation de l'enregistrement, vous devez arrêter toutes les tâches en cours d'exécution sur l'instance de conteneur avant l'annulation de l'enregistrement. Vous éviterez ainsi que les tâches orphelines consomment des ressources.

Lors de la désinscription d'une instance de conteneur, tenez compte des considérations suivantes.

- Étant donné que chaque instance de conteneur possède des informations d'état uniques, elles ne peuvent pas faire l'objet d'une annulation d'enregistrement dans un cluster et d'un réenregistrement dans un autre. Pour déplacer des ressources d'instance de conteneur, nous vous recommandons de résilier les instances de conteneur d'un cluster et d'en lancer de nouvelles dans le nouveau cluster. Pour plus d'informations, consultez Résilier votre instance dans le guide de EC2 l'utilisateur Amazon etLancement d'une instance de conteneur Amazon ECS Linux.

- Si l'instance de conteneur est gérée par un groupe ou une AWS CloudFormation pile Auto Scaling, mettez fin à l'instance en mettant à jour le groupe ou la AWS CloudFormation pile Auto Scaling. Sinon, le groupe Auto Scaling ou AWS CloudFormation créera une instance après sa résiliation.
- Si vous résiliez une instance de conteneur en cours d'exécution avec un agent de conteneur Amazon ECS connecté, l'agent annule automatiquement l'enregistrement de l'instance dans votre cluster. L'annulation de l'enregistrement des instances de conteneur arrêtées ou des instances ayant des agents déconnectés n'est pas automatique lorsque ces instances sont résiliées.
- Le désenregistrement d'une instance de conteneur supprime l'instance d'un cluster, mais ne met pas fin à l'instance Amazon. EC2 Si vous avez fini d'utiliser l'instance, veillez à la résilier pour arrêter la facturation. Pour plus d'informations, consultez la section Résiliation de votre instance dans le guide de EC2 l'utilisateur Amazon.

Procédure

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, choisissez la région dans laquelle votre instance externe est inscrite.
3. Dans le panneau de navigation, choisissez Clusters, puis sélectionnez le cluster qui héberge l'instance.
4. Sur la *name* page Cluster :, choisissez l'onglet Infrastructure.
5. Sous Constainer instances (Instances de conteneur), sélectionnez l'ID d'instance pour annuler l'enregistrement. Vous êtes redirigé vers la page de détails de l'instance de conteneur.
6. Sur la *id* page Container Instance :, choisissez Désenregistrer.
7. Dans le message de confirmation, Annuler l'inscription.
8. Si vous en avez terminé avec l'instance de conteneur, mettez fin à l' EC2instance Amazon sous-jacente. Pour plus d'informations, consultez Résilier votre instance dans le guide de EC2 l'utilisateur Amazon.

Vidange des instances de conteneurs Amazon ECS

Il peut arriver que vous deviez supprimer une instance de conteneur de votre cluster, par exemple pour effectuer des mises à jour du système ou pour réduire la capacité du cluster. Amazon ECS offre la possibilité de transférer une instance de conteneur vers un état `DRAINING`. C'est ce que l'on

appelle le drainage des instances de conteneur. Lorsqu'une instance de conteneur est définie sur `DRAINING`, Amazon ECS bloque la planification du placement des nouvelles tâches sur l'instance de conteneur.

Comportement de drainage pour les services

Toutes les tâches faisant partie d'un service dans un état `PENDING` sont arrêtées immédiatement. S'il existe une capacité d'instance de conteneur disponible dans le cluster, le planificateur de service va démarrer les tâches de remplacement. S'il n'y a pas assez de capacité d'instance de conteneur, un message d'événement de service sera envoyé pour indiquer le problème.

Les tâches faisant partie d'un service sur l'instance de conteneur qui se trouvent à l'état `RUNNING` passent à l'état `STOPPED`. Le planificateur de service tente de remplacer les tâches en fonction du type de déploiement du service et des paramètres de configuration du déploiement, `minimumHealthyPercent` et `maximumPercent`. Pour plus d'informations, consultez Services Amazon ECS et Paramètres de définition du service Amazon ECS.

- Si `minimumHealthyPercent` est inférieur à 100 %, le planificateur peut ignorer `desiredCount` temporairement pendant le remplacement de tâche. Par exemple, si le `desiredCount` est de quatre tâches, un minimum de 50 % permet au planificateur d'arrêter deux tâches existantes avant de commencer deux nouvelles tâches. Si le minimum est de 100 %, le planificateur de service ne peut pas supprimer les tâches existantes tant que les tâches de remplacement sont considérées comme saines. Si les tâches des services qui n'utilisent pas d'équilibreur de charge ont l'état `RUNNING`, elles sont considérées comme saines. Les tâches des services qui utilisent un équilibreur de charge sont considérées comme saines si elles se trouvent à l'état `RUNNING` et si l'instance de conteneur sur laquelle elles sont hébergées est signalée comme saine par l'équilibreur de charge.

  > **Important**
  >
  > Si vous utilisez des instances Spot et que la valeur `minimumHealthyPercent` est supérieure ou égale à 100 %, le service ne disposera pas de suffisamment de temps pour remplacer la tâche avant la fin de l'instance Spot.

- Le paramètre `maximumPercent` représente une limite supérieure du nombre de tâches en cours d'exécution lors du remplacement des tâches, ce qui vous permet de définir la taille du lot de remplacement. Par exemple, si le `desiredCount` est de quatre tâches, un maximum de 200 % lance quatre nouvelles tâches avant d'arrêter les quatre tâches à drainer (à condition de disposer

des ressources de cluster nécessaires). Si le maximum est de 100 %, les tâches de remplacement ne peuvent pas commencer tant que les tâches de drainage sont arrêtées.

> **Important**
>
> Si `minimumHealthyPercent` et `maximumPercent` sont à 100 %, alors le service ne peut pas supprimer les tâches existantes et ne peut pas non plus démarrer les tâches de remplacement. Cela empêche le drainage des instances de conteneur ainsi que d'effectuer de nouveaux déploiements.

Comportement de drainage pour les tâches autonomes

Toutes les tâches autonomes à l'état `PENDING` ou `RUNNING` ne sont pas affectées. Vous devez attendre qu'elles s'arrêtent d'elles-mêmes ou les arrêter manuellement. L'instance de conteneur restera dans l'état `DRAINING`.

Une instance de conteneur a terminé le drainage lorsque toutes les tâches qui s'exécutent sur l'instance passent à l'état `STOPPED`. L'instance de conteneur reste à l'état `DRAINING` jusqu'à ce qu'elle soit réactivée ou supprimée. Vous pouvez vérifier l'état des tâches sur l'instance de conteneur en utilisant l'ListTasksopération avec le `containerInstance` paramètre pour obtenir une liste des tâches de l'instance, suivie d'une DescribeTasksopération avec le nom de ressource Amazon (ARN) ou l'ID de chaque tâche pour vérifier l'état de la tâche.

Lorsque vous êtes prêt pour que l'instance de conteneur recommence à héberger des tâches, vous modifiez l'état de l'instance de conteneur de `DRAINING` en `ACTIVE`. Le planificateur Amazon ECS service considérera ensuite à nouveau l'instance de conteneur pour le placement des tâches.

Procédure

Les étapes suivantes peuvent être utilisées pour définir une instance de conteneur à drainer à l'aide de la nouvelle AWS Management Console.

Vous pouvez également utiliser l'action ou la update-container-instances-statecommande de l'UpdateContainerInstancesStateAPI pour modifier le statut d'une instance de conteneur en`DRAINING`.

AWS Management Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.

2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez un cluster qui héberge vos instances.
4. Sur la *name* page Cluster :, choisissez l'onglet Infrastructure. Puis, sous Container instances (Instances de conteneur), cochez la case correspondant à chaque instance de conteneur que vous souhaitez drainer.
5. Choisissez Actions, Drain (Drainer).

Instances de EC2 conteneurs Amazon ECS

L'agent Amazon ECS est un processus qui s'exécute sur chaque instance de conteneur enregistrée auprès de votre cluster. Il facilite la communication entre vos instances de conteneur et Amazon ECS.

> **Note**
>
> Sur les instances de conteneur Linux, le conteneur d'agents monte des répertoires de premier niveau tels que `/lib/lib64`, et. `/proc` Cela est nécessaire pour les fonctionnalités d'ECS telles que les volumes Amazon EBS, le mode `awsvpc` réseau, Amazon ECS Service Connect et FireLens Amazon ECS.

Chaque version d'agent de conteneur Amazon ECS prend en charge un ensemble de différentes fonctions et fournit des correctifs des versions précédentes. Dans la mesure du possible, nous recommandons toujours d'utiliser la dernière version de l'agent de conteneur Amazon ECS. Pour mettre à jour votre agent de conteneur avec la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

L'agent de conteneur Amazon ECS contient l'`amazon-ecs-pause`image. Amazon ECS utilise cette image pour les tâches qui utilisent le mode `awsvpc` réseau.

Pour voir quelles fonctionnalités et améliorations sont incluses dans chaque version de l'agent, consultez https://github.com/aws/amazon-ecs-agent/releases.

> **Important**
>
> La version minimale de Docker pour des métriques fiables est la version `v20.10.13` et les versions ultérieures, qui sont incluses dans l'AMI `20220607` optimisée pour Amazon ECS et les versions plus récentes.
> La version de l'agent Amazon ECS `1.20.0` et les versions plus récentes ne prennent plus en charge les versions de Docker antérieures à `18.01.0`.

Cycle de vie

Lorsque l'agent de conteneur Amazon ECS enregistre une EC2 instance Amazon dans votre cluster, l' EC2instance Amazon indique son statut en tant que `ACTIVE` et son état de connexion à l'agent en tant que`TRUE`. Cette instance de conteneur peut exécuter des demandes de tâches.

Si vous arrêtez (sans la résilier) une instance de conteneur, son statut reste `ACTIVE`, mais le statut de connexion de l'agent devient `FALSE` en quelques minutes. Toutes les tâches qui étaient en cours d'exécution sur l'instance de conteneur s'arrêtent. Si vous redémarrez l'instance de conteneur, l'agent de conteneur se connecte à nouveau avec le service Amazon ECS service et vous êtes à nouveau en mesure d'exécuter des tâches sur l'instance.

> **Important**
>
> Si vous arrêtez et démarrez une instance de conteneur, ou si vous redémarrez cette instance, certaines versions antérieures de l'agent de conteneur Amazon ECS enregistrent à nouveau l'instance sans annuler l'enregistrement de l'ID d'instance de conteneur d'origine. C'est pourquoi Amazon ECS indique un nombre d'instances de conteneur dans votre cluster supérieur au nombre réel. (Si vous avez une instance de conteneur dupliquée IDs pour le même ID d' EC2 instance Amazon, vous pouvez désenregistrer en toute sécurité les doublons répertoriés comme `ACTIVE` ayant un statut de connexion à l'agent de.) `FALSE` Ce problème est corrigé dans la version actuelle de l'agent de conteneur Amazon ECS. Pour plus d'informations sur la mise à jour de la version actuelle, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Si vous remplacez le statut d'une instance de conteneur par `DRAINING`, les nouvelles tâches ne sont pas placées sur l'instance de conteneur. Toutes les tâches de service en cours d'exécution sur l'instance de conteneur sont supprimées, si possible, afin que vous puissiez effectuer les mises

à jour du système. Pour de plus amples informations, veuillez consulter Vidange des instances de conteneurs Amazon ECS.

Si vous annulez l'enregistrement d'une instance de conteneur ou la résiliez, le statut de l'instance de conteneur passe immédiatement à `INACTIVE` et l'instance de conteneur ne figure plus dans la liste des instances de conteneur. Cependant, vous pouvez toujours décrire l'instance de conteneur pendant une heure après la résiliation. Après un délai d'une heure, la description de l'instance n'est plus disponible.

> **Important**
>
> Vous pouvez drainer les instances manuellement ou créer un hook de cycle de vie du groupe Auto Scaling pour définir l'état de l'instance sur `DRAINING`. Consultez Amazon EC2 Auto Scaling Lifecycle Hooks pour plus d'informations sur les hooks du cycle de vie Auto Scaling.

Support Docker

Amazon ECS prend en charge les deux dernières versions majeures de Docker publiées sur Amazon Linux. Actuellement, cela inclut Docker 20.10.x et Docker 25.x.

La version minimale requise de Docker pour Amazon ECS se trouve dans le fichier de spécification de l'agent Amazon ECS sur GitHub.

Lorsque vous utilisez l'AMI optimisée pour Amazon ECS, Docker est préinstallé et configuré pour fonctionner avec l'agent de conteneur Amazon ECS. L'AMI inclut une version de Docker testée et prise en charge par Amazon ECS.

> **Note**
>
> Amazon ECS prend en charge plusieurs versions de Docker, mais nous vous recommandons d'utiliser la version Docker fournie avec l'AMI optimisée pour Amazon ECS pour une compatibilité et un support optimaux.

AMI optimisée pour Amazon ECS

Les variantes Linux de l'AMI optimisée pour Amazon ECS utilisent l'AMI Amazon Linux 2 comme base. Le nom de l'AMI source Amazon Linux 2 pour chaque variante peut être récupéré en

interrogeant l'API de Systems Manager Parameter Store. Pour de plus amples informations, veuillez consulter Récupération des métadonnées de l'AMI Linux optimisées pour Amazon ECS. Lorsque vous lancez nos instances de conteneur à partir de l'AMI Amazon Linux 2 optimisée pour Amazon ECS la plus récente, vous recevez la version actuelle de l'agent de conteneur. Pour lancer une instance de conteneur avec la dernière AMI Amazon Linux 2 optimisée pour Amazon ECS, veuillez consulter Lancement d'une instance de conteneur Amazon ECS Linux.

Informations supplémentaires

Les pages suivantes fournissent des informations supplémentaires sur les modifications :

- Connectez-vous à la liste des modifications apportées à l'agent Amazon ECS GitHub
- Le code source de l'application `ecs-init`, ainsi que les scripts et la configuration nécessaires à l'empaquetage de l'agent, font désormais partie du référentiel de l'agent. Pour les anciennes versions `ecs-init` et emballages, consultez le journal des modifications Amazon ecs-init sur GitHub
- Notes de mise à jour pour Amazon Linux 2.
- Notes de mise à jour Docker Engine dans la documentation Docker
- Documentation du pilote NVIDIA dans la documentation NVIDIA

Configuration de l'agent de conteneur Amazon ECS

S'applique à : EC2 instances

L'agent de conteneur Amazon ECS prend en charge un certain nombre d'options de configuration, dont la plupart sont définies par le biais de variables d'environnement.

Si votre instance de conteneur a été lancée avec une variante Linux de l'AMI optimisée pour Amazon ECS, vous pouvez définir ces variables d'environnement dans le fichier `/etc/ecs/ecs.config`, puis redémarrer l'agent. Vous pouvez également écrire ces variables de configuration dans vos instances de conteneur avec les données EC2 utilisateur Amazon au moment du lancement. Pour de plus amples informations, veuillez consulter Démarrage des instances de conteneur Linux Amazon ECS pour transmettre des données.

Si votre instance de conteneur a été lancée avec une variante Windows de l'AMI optimisée pour Amazon ECS, vous pouvez définir ces variables d'environnement à l'aide de la PowerShell SetEnvironmentVariable commande, puis redémarrer l'agent. Pour plus d'informations, consultez Exécuter des commandes lorsque vous lancez une EC2 instance avec des données utilisateur

saisies dans le guide de EC2 l'utilisateur Amazon etthe section called “Démarrage des instances de conteneur”.

Si vous démarrez manuellement l'agent de conteneur Amazon ECS (pour les applications non optimisées pour Amazon ECS AMIs), vous pouvez utiliser ces variables d'environnement dans la docker run commande que vous utilisez pour démarrer l'agent. Utilisez ces variables avec la syntaxe `--env=VARIABLE_NAME=VARIABLE_VALUE`. Pour les informations sensibles, telles que les informations d'authentification des référentiels privés, vous devez stocker vos variables d'environnement d'agent dans un fichier et les transmettre toutes à la fois avec l'option `--env-file path_to_env_file`. Vous pouvez utiliser les commandes suivantes pour ajouter les variables.

```
sudo systemctl stop ecs
sudo vi /etc/ecs/ecs.config
# And add the environment variables with VARIABLE_NAME=VARIABLE_VALUE format.
sudo systemctl start ecs
```

Exécutez l'agent Amazon ECS avec l'espace de noms PID de l'hôte

Par défaut, l'agent Amazon ECS s'exécute avec son propre espace de noms PID. Dans les configurations suivantes, vous pouvez configurer l'agent Amazon ECS pour qu'il s'exécute avec l'espace de noms PID de l'hôte :

- SELinux le mode d'application est activé.
- La politique de SELinux sécurité de Docker est définie sur true.

Vous pouvez configurer ce comportement en définissant la variable d'`ECS_AGENT_PID_NAMESPACE_HOST`environnement sur `true` dans votre `/etc/ecs/ecs.config` fichier. Lorsque cette variable est activée, le conteneur de l'agent Amazon ECS `ecs-init` démarre avec l'espace de noms PID de l'hôte (`--pid=host`), ce qui permet à l'agent de s'amorcer correctement dans SELinux les environnements d'application. Cette fonctionnalité est disponible dans les versions de l'agent Amazon ECS `1.94.0` et versions ultérieures.

Pour activer cette fonctionnalité, ajoutez la ligne suivante à votre `/etc/ecs/ecs.config` fichier :

```
ECS_AGENT_PID_NAMESPACE_HOST=true
```

Après avoir effectué cette modification, redémarrez l'agent Amazon ECS pour que la modification soit prise en compte :

```
sudo systemctl restart ecs
```

Les fonctionnalités suivantes ne fonctionneront pas : le mode d' SELinux exécution est activé et la politique de sécurité Docker est définie sur true, même lorsqu'elle `ECS_AGENT_PID_NAMESPACE_HOST=true` est définie.

- Amazon ECS Exec
- Joindre une tâche Amazon EBS
- Service Connect
- FireLenspour Amazon ECS

Paramètres disponibles

Pour plus d'informations sur les paramètres de configuration de l'agent de conteneur Amazon ECS disponibles, consultez Amazon ECS Container Agent on GitHub.

Stockage de la configuration de l'instance de conteneur Amazon ECS dans Amazon S3

La configuration de l'agent de conteneur Amazon ECS est contrôlée à l'aide de la variable d'environnement. Les variantes Linux de l'AMI optimisée pour Amazon ECS vérifient la présence de ces variables dans `/etc/ecs/ecs.config` lorsque l'agent de conteneur démarre et configurent l'agent en conséquence. Des variables d'environnement inutiles, telles que`ECS_CLUSTER`, peuvent être transmises à l'instance de conteneur lors du lancement via les données EC2 utilisateur d'Amazon et écrites dans ce fichier sans conséquence. Cependant, d'autres informations sensibles, telles que vos AWS informations d'identification ou la `ECS_ENGINE_AUTH_DATA` variable, ne doivent jamais être transmises à une instance `/etc/ecs/ecs.config` dans les données utilisateur ou écrites de manière à ce qu'elles apparaissent dans un `.bash_history` fichier.

Stocker les informations de configuration dans un compartiment privé d'Amazon S3 et accorder un accès en lecture seule au rôle IAM de votre instance de conteneur est un moyen pratique et sécurisé d'autoriser la configuration d'une instance de conteneur au moment du lancement. Vous pouvez stocker une copie de votre fichier `ecs.config` dans un compartiment privé. Vous pouvez ensuite utiliser les données EC2 utilisateur Amazon pour installer AWS CLI et copier vos informations de configuration au `/etc/ecs/ecs.config` moment du lancement de l'instance.

Pour stocker un fichier **`ecs.config`** dans Amazon S3

1. Vous devez accorder au rôle d'instance de conteneur (ecsInstanceRole) des autorisations pour avoir un accès en lecture seule à Amazon S3. Vous pouvez le faire en assignant l'AmazonS3 ReadOnlyAccess au rôle. `ecsInstanceRole` Pour plus d'informations sur la façon d'associer une politique à un rôle, voir Mettre à jour les autorisations pour un rôle dans le Guide de AWS Identity and Access Management l'utilisateur
2. Créez un fichier `ecs.config` contenant des variables de configuration d'agent Amazon ECS valides en utilisant le format suivant. Cet exemple configure l'authentification de registre privé. Pour de plus amples informations, veuillez consulter Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.

```
ECS_ENGINE_AUTH_TYPE=dockercfg
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example.com"}}
```

> **Note**
>
> Pour obtenir la liste complète des variables de configuration des agents Amazon ECS disponibles, consultez Amazon ECS Container Agent on GitHub.

3. Pour stocker votre fichier de configuration, créez un compartiment privé dans Amazon S3. Pour plus d'informations, consultez la section Création d'un compartiment dans le guide de l'utilisateur d'Amazon Simple Storage Service.
4. Chargez le fichier `ecs.config` dans votre compartiment S3. Pour plus d'informations, consultez Chargement d'objets dans le Guide de l'utilisateur Amazon Simple Storage Service.

Pour charger un fichier **`ecs.config`** à partir d'Amazon S3 lors du lancement

1. Exécutez les précédentes procédures de la présente section pour autoriser l'accès en lecture seule d'Amazon S3 à vos instances de conteneur et stocker un fichier `ecs.config` dans un compartiment S3 privé.
2. Lancez de nouvelles instances de conteneur et utilisez l'exemple de script suivant dans les données EC2 utilisateur. Le script installe AWS CLI et copie votre fichier de configuration dans. `/etc/ecs/ecs.config` Pour de plus amples informations, veuillez consulter Lancement d'une instance de conteneur Amazon ECS Linux.

```
#!/bin/bash
yum install -y aws-cli
aws s3 cp s3://your_bucket_name/ecs.config /etc/ecs/ecs.config
```

Installation de l'agent de conteneur Amazon ECS

Si vous souhaitez enregistrer une EC2 instance Amazon auprès de votre cluster Amazon ECS et que cette instance n'utilise pas d'AMI basée sur l'AMI optimisée pour Amazon ECS, vous pouvez installer l'agent de conteneur Amazon ECS manuellement en suivant la procédure suivante. Pour ce faire, vous pouvez télécharger l'agent depuis l'un des compartiments Amazon S3 régionaux ou depuis Amazon Elastic Container Registry Public. Si vous effectuez le téléchargement depuis l'un des compartiments Amazon S3 régionaux, vous pouvez éventuellement vérifier la validité du fichier d'agent de conteneur à l'aide de la signature PGP.

> **Note**
>
> Les unités `systemd` pour les services Amazon ECS et Docker sont soumises à une directive qui demande d'attendre que `cloud-init` ait terminé avant de démarrer les deux services. Le `cloud-init` processus n'est pas considéré comme terminé tant que l'exécution de vos données EC2 utilisateur Amazon n'est pas terminée. Par conséquent, le démarrage d'Amazon ECS ou Docker via les données EC2 utilisateur d'Amazon peut provoquer un blocage. Pour démarrer l'agent de conteneur à l'aide des données EC2 utilisateur Amazon, vous pouvez utiliser`systemctl enable --now --no-block ecs.service`.

Installation de l'agent de conteneur Amazon ECS sur une instance autre que l'Amazon Linux EC2 instance

Pour installer l'agent de conteneur Amazon ECS sur une EC2 instance Amazon, vous pouvez télécharger l'agent depuis l'un des compartiments Amazon S3 régionaux et l'installer.

> **Note**
>
> Lorsque vous utilisez une AMI autre qu'Amazon Linux, votre EC2 instance Amazon a besoin d'une `cgroupfs` assistance pour le `cgroup` pilote afin que l'agent Amazon ECS puisse

prendre en charge les limites de ressources au niveau des tâches. Pour plus d'informations, consultez la section Agent Amazon ECS sur GitHub.

Les fichiers de l'agent de conteneur Amazon ECS le plus récent sont répertoriés par région ci-dessous à des fins de référence pour chaque architecture système.

Région	Nom de la région	Fichiers deb init Amazon ECS	Fichiers rpm init Amazon ECS
us-east-2	USA Est (Ohio)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-east-1	USA Est (Virginie du Nord)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-west-1	USA Ouest (Californie du Nord)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-west-2	USA Ouest (Oregon)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-east-1	Asie-Pacifique (Hong Kong)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Région	Nom de la région	Fichiers deb init Amazon ECS	Fichiers rpm init Amazon ECS
ap-northeast-1	Asie-Pacifique (Tokyo)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-northeast-2	Asie-Pacifique (Séoul)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-south-1	Asie-Pacifique (Mumbai)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-southeast-1	Asie-Pacifique (Singapour)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-southeast-2	Asie-Pacifique (Sydney)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ca-central-1	Canada (Centre)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Région	Nom de la région	Fichiers deb init Amazon ECS	Fichiers rpm init Amazon ECS
eu-central-1	Europe (Francfort)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-west-1	Europe (Irlande)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-west-2	Europe (Londres)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-west-3	Europe (Paris)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
sa-east-1	Amérique du Sud (São Paulo)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 Amazon ECS init aarch64 (aarch64)
us-gov-east-1	AWS GovCloud (USA Est)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Région	Nom de la région	Fichiers deb init Amazon ECS	Fichiers rpm init Amazon ECS
us-gov-west-1	AWS GovCloud (US-Ouest)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Pour installer l'agent de conteneur Amazon ECS sur une EC2 instance Amazon à l'aide d'une Amazon Linux non-AMI

1. Lancez une EC2 instance Amazon avec un rôle IAM qui permet d'accéder à Amazon ECS. Pour de plus amples informations, veuillez consulter Rôle IAM d'instance de conteneur Amazon ECS.
2. Connectez-vous à votre instance.
3. Installez la dernière version de Docker sur votre instance.
4. Vérifiez votre version de Docker pour vous assurer que votre système répond à l'exigence de version minimale.

   > **Note**
   >
   > La version minimale de Docker pour des métriques fiables est la version `v20.10.13` et les versions ultérieures, qui sont incluses dans l'AMI `20220607` optimisée pour Amazon ECS et les versions plus récentes.
   > La version de l'agent Amazon ECS `1.20.0` et les versions plus récentes ne prennent plus en charge les versions de Docker antérieures à `18.01.0`.

   ```
   docker --version
   ```

5. Téléchargez le fichier d'agent Amazon ECS approprié pour votre système d'exploitation et votre architecture système, puis installez-le.

   Pour les architectures `deb` :

   ```
   ubuntu:~$ curl -O https://s3.us-west-2.amazonaws.com/amazon-ecs-agent-us-west-2/
   amazon-ecs-init-latest.amd64.deb
   ```

```
ubuntu:~$ sudo dpkg -i amazon-ecs-init-latest.amd64.deb
```

Pour les architectures `rpm` :

```
fedora:~$ curl -O https://s3.us-west-2.amazonaws.com/amazon-ecs-agent-us-west-2/
amazon-ecs-init-latest.x86_64.rpm
fedora:~$ sudo yum localinstall -y amazon-ecs-init-latest.x86_64.rpm
```

6. Modifiez le `/lib/systemd/system/ecs.service` fichier et ajoutez la ligne suivante à la fin de la `[Unit]` section.

```
After=cloud-final.service
```

7. (Facultatif) Pour enregistrer l'instance auprès d'un cluster autre que le cluster `default`, modifiez le fichier `/etc/ecs/ecs.config` et ajoutez le contenu suivant. L'exemple suivant spécifie le cluster `MyCluster` :

```
ECS_CLUSTER=MyCluster
```

Pour plus d'informations sur ces options ainsi que d'autres options d'exécution d'agents, consultez la page Configuration de l'agent de conteneur Amazon ECS.

> **Note**
>
> Vous pouvez éventuellement stocker les variables d'environnement de votre agent dans Amazon S3 (qui peuvent être téléchargées sur vos instances de conteneur au moment du lancement à l'aide des données EC2 utilisateur Amazon). Ceci est particulièrement conseillé pour les informations sensibles, telles que les informations d'authentification pour les référentiels privés. Pour plus d'informations, consultez Stockage de la configuration de l'instance de conteneur Amazon ECS dans Amazon S3 et Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.

8. Lancez le service `ecs`.

```
ubuntu:~$ sudo systemctl start ecs
```

Exécution de l'agent Amazon ECS avec le mode réseau hôte

Lorsque vous exécutez l'agent de conteneur Amazon ECS, `ecs-init` créera le conteneur d'agent de conteneur avec le mode de réseau `host`. C'est le seul mode de réseau pris en charge pour le conteneur d'agent de conteneur.

Cela vous permet de bloquer l'accès au point de terminaison du service de métadonnées d' EC2 instance Amazon (`http://169.254.169.254`) pour les conteneurs démarrés par l'agent de conteneur. Cela permet d'assurer que les conteneurs ne sont pas en mesure d'accéder aux informations d'identification du rôle IAM à partir du profil d'instance de conteneur et impose que les tâches utilisent uniquement les informations d'identification de rôle de tâche IAM. Pour de plus amples informations, veuillez consulter Rôle IAM de la tâche Amazon ECS.

Cela permet également à l'agent de conteneur de ne pas devoir disputer les connexions et le trafic réseau sur la passerelle `docker0`.

Paramètres de configuration du journal de l'agent de conteneur Amazon ECS

L'agent de conteneur Amazon ECS stocke les journaux sur vos instances de conteneur.

Pour l'agent de conteneur version 1.36.0 et ultérieure, les journaux sont par défaut situés sur `/var/log/ecs/ecs-agent.log` sur les instances Linux et `C:\ProgramData\Amazon\ECS\log\ecs-agent.log` sur les instances Windows.

Pour l'agent de conteneur version 1.35.0 et antérieure, les journaux sont par défaut situés sur `/var/log/ecs/ecs-agent.log.`*`timestamp`* sur les instances Linux et `C:\ProgramData\Amazon\ECS\log\ecs-agent.log.`*`timestamp`* sur les instances Windows.

Par défaut, les journaux de l'agent font l'objet d'une rotation toutes les heures et un maximum de 24 journaux sont stockés.

Voici les variables de configuration de l'agent de conteneur qui peuvent être utilisées pour modifier le comportement de journalisation de l'agent par défaut. Pour obtenir des informations détaillées sur tous les paramètres de configuration disponibles, consultez Configuration de l'agent de conteneur Amazon ECS le fichier README de l'agent Amazon ECS sur GitHub.

Pour l'agent de conteneur version 1.36.0 et ultérieure, voici un exemple de fichier journal lorsque le format `logfmt` est utilisé.

```
level=info time=2019-12-12T23:43:29Z msg="Loading configuration" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Image excluded from cleanup: amazon/amazon-
ecs-agent:latest" module=parse.go
level=info time=2019-12-12T23:43:29Z msg="Image excluded from cleanup: amazon/amazon-
ecs-pause:0.1.0" module=parse.go
level=info time=2019-12-12T23:43:29Z msg="Amazon ECS agent Version: 1.36.0, Commit:
 ca640387" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Creating root ecs cgroup: /ecs"
 module=init_linux.go
level=info time=2019-12-12T23:43:29Z msg="Creating cgroup /ecs"
 module=cgroup_controller_linux.go
level=info time=2019-12-12T23:43:29Z msg="Loading state!" module=statemanager.go
level=info time=2019-12-12T23:43:29Z msg="Event stream ContainerChange start
 listening..." module=eventstream.go
level=info time=2019-12-12T23:43:29Z msg="Restored cluster 'auto-robc'" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Restored from checkpoint file. I
 am running as 'arn:aws:ecs:us-west-2:0123456789:container-instance/auto-
robc/3330a8a91d15464ea30662d5840164cd' in cluster 'auto-robc'" module=agent.go
```

Voici un exemple de fichier journal lorsque le format JSON est utilisé.

```
{"time": "2019-11-07T22:52:02Z", "level": "info", "msg": "Starting Amazon Elastic
 Container Service Agent", "module": "engine.go"}
```

Pour les versions 1.35.0 et antérieures de l'agent de conteneur, voici le format du fichier journal.

```
2016-08-15T15:54:41Z [INFO] Starting Agent: Amazon ECS Agent - v1.12.0 (895f3c1)
2016-08-15T15:54:41Z [INFO] Loading configuration
2016-08-15T15:54:41Z [WARN] Invalid value for task cleanup duration, will be overridden
 to 3h0m0s, parsed value 0, minimum threshold 1m0s
2016-08-15T15:54:41Z [INFO] Checkpointing is enabled. Attempting to load state
2016-08-15T15:54:41Z [INFO] Loading state! module="statemanager"
2016-08-15T15:54:41Z [INFO] Detected Docker versions [1.17 1.18 1.19 1.20 1.21 1.22]
2016-08-15T15:54:41Z [INFO] Registering Instance with ECS
2016-08-15T15:54:41Z [INFO] Registered! module="api client"
```

Configuration des instances de conteneur Amazon ECS pour les images Docker privées

L'agent de conteneur Amazon ECS peut authentifier au moyen de registres privés, à l'aide de l'authentification de base. Lorsque vous activez l'authentification de registres privés, vous pouvez

utiliser des images Docker privées dans vos définitions de tâche. Cette fonction est prise en charge uniquement pour les tâches lorsque vous utilisez le type de lancement EC2 .

Une autre méthode d'activation de l'authentification du registre privé consiste AWS Secrets Manager à stocker vos informations d'identification de registre privé en toute sécurité, puis à les référencer dans la définition de votre conteneur. Cela permet à vos tâches d'utiliser des images de référentiels privés. Cette méthode prend en charge les tâches utilisant le type de lancement Fargate EC2 ou le type de lancement Fargate. Pour de plus amples informations, veuillez consulter Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.

Au lancement, l'agent de conteneur Amazon ECS recherche deux variables d'environnement :

- `ECS_ENGINE_AUTH_TYPE`, qui spécifie le type des données d'authentification envoyées.
- `ECS_ENGINE_AUTH_DATA`, qui contient les informations d'authentification réelles.

Les variantes Linux de l'AMI optimisée pour Amazon ECS analysent le `/etc/ecs/ecs.config` fichier à la recherche de ces variables au lancement de l'instance de conteneur et à chaque démarrage du service (avec la sudo start ecs commande). AMIs qui ne sont pas optimisés pour Amazon ECS doivent stocker ces variables d'environnement dans un fichier et les transmettre avec l'`--env-file` *`path_to_env_file`*option à la docker run commande qui démarre l'agent de conteneur.

> **Important**
>
> Nous vous déconseillons d'injecter ces variables d'environnement d'authentification au lancement de l'instance avec les données EC2 utilisateur Amazon ou de les transmettre avec l'`--env`option à la docker run commande. Ces méthodes ne sont pas appropriées pour les données sensibles, par exemple les informations d'authentification. Pour plus d'informations sur l'ajout en toute sécurité d'informations d'authentification à vos instances de conteneur, consultez Stockage de la configuration de l'instance de conteneur Amazon ECS dans Amazon S3.

Formats d'authentification

Il existe deux formats disponibles pour une authentification de registre privé, `dockercfg` et `docker`.

Format d'authentification dockercfg

Le format `dockercfg` utilise les informations d'authentification stockées dans le fichier de configuration qui est créé lorsque vous exécutez la commande docker login. Pour créer ce fichier, exécutez docker login sur votre système local et saisissez votre nom d'utilisateur de registre, votre mot de passe et une adresse e-mail. Vous pouvez également vous connecter à une instance de conteneur et y exécuter la commande. Selon votre version de Docker, ce fichier est enregistré sous la forme `~/.dockercfg` ou `~/.docker/config.json`.

```
cat ~/.docker/config.json
```

Sortie :

```
{
  "auths": {
    "https://index.docker.io/v1/": {
      "auth": "zq212MzEXAMPLE7o6T25Dk0i"
    }
  }
}
```

> **Important**
>
> Les versions les plus récentes de Docker créent un fichier de configuration comme illustré ci-après avec un objet `auths` externe. L'agent Amazon ECS prend en charge les données d'authentification `dockercfg` au format ci-dessous uniquement, sans l'objet `auths`. Si l'utilitaire jq est installé, vous pouvez extraire ces données à l'aide de la commande suivante : cat ~/.docker/config.json | jq .auths

```
cat ~/.docker/config.json | jq .auths
```

Sortie :

```
{
  "https://index.docker.io/v1/": {
    "auth": "zq212MzEXAMPLE7o6T25Dk0i",
    "email": "email@example.com"
  }
}
```

Dans l'exemple ci-dessus, les variables d'environnement suivantes doivent être ajoutées au fichier de variables d'environnement (`/etc/ecs/ecs.config` pour l'AMI optimisée pour Amazon ECS) que l'agent de conteneur Amazon ECS charge au moment de l'exécution. Si vous n'utilisez pas l'AMI optimisée pour Amazon ECS et que vous démarrez l'agent manuellement avec docker run, spécifiez le fichier de variables d'environnement avec l'option `--env-file` *`path_to_env_file`* lorsque vous démarrez l'agent.

```
ECS_ENGINE_AUTH_TYPE=dockercfg
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example.com"}}
```

Vous pouvez configurer plusieurs registres privés avec la syntaxe suivante :

```
ECS_ENGINE_AUTH_TYPE=dockercfg
ECS_ENGINE_AUTH_DATA={"repo.example-01.com":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example-01.com"},"repo.example-02.com":
{"auth":"fQ172MzEXAMPLEoF7225DU0j","email":"email@example-02.com"}}
```

Format d'authentification docker

Le format `docker` utilise une représentation JSON du serveur de registre que l'agent doit utiliser pour l'authentification. Il inclut également les paramètres d'authentification requis par ce registre (par exemple, nom d'utilisateur, mot de passe et adresse e-mail pour ce compte). Pour un compte Docker Hub, la représentation JSON se présente comme suit :

```
{
  "https://index.docker.io/v1/": {
    "username": "my_name",
    "password": "my_password",
    "email": "email@example.com"
  }
}
```

Dans cet exemple, les variables d'environnement suivantes doivent être ajoutées au fichier de variables d'environnement (`/etc/ecs/ecs.config` pour l'AMI optimisée pour Amazon ECS) que l'agent de conteneur Amazon ECS charge au moment de l'exécution. Si vous n'utilisez pas l'AMI optimisée pour Amazon ECS et que vous démarrez l'agent manuellement avec docker run, spécifiez le fichier de variables d'environnement avec l'option `--env-file` *`path_to_env_file`* lorsque vous démarrez l'agent.

```
ECS_ENGINE_AUTH_TYPE=docker
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"username":"my_name","password":"my_password","email":"email@example.com"}}
```

Vous pouvez configurer plusieurs registres privés avec la syntaxe suivante :

```
ECS_ENGINE_AUTH_TYPE=docker
ECS_ENGINE_AUTH_DATA={"repo.example-01.com":
{"username":"my_name","password":"my_password","email":"email@example-01.com"},"repo.example-02
{"username":"another_name","password":"another_password","email":"email@example-02.com"}}
```

Procédure

Utilisez la procédure suivante pour activer les registres privés pour vos instances de conteneur.

Pour activer les registres privés dans l'AMI optimisée pour Amazon ECS

1. Connectez-vous à votre instance de conteneur à l'aide de SSH.
2. Ouvrez le fichier `/etc/ecs/ecs.config` et ajoutez les valeurs `ECS_ENGINE_AUTH_TYPE` et `ECS_ENGINE_AUTH_DATA` pour votre registre et votre compte :

   ```
   sudo vi /etc/ecs/ecs.config
   ```

   Cet exemple authentifie un compte d'utilisateur Docker Hub :

   ```
   ECS_ENGINE_AUTH_TYPE=docker
   ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
   {"username":"my_name","password":"my_password","email":"email@example.com"}}
   ```

3. Vérifiez si votre agent utilise la variable d'environnement `ECS_DATADIR` pour enregistrer son état :

   ```
   docker inspect ecs-agent | grep ECS_DATADIR
   ```

   Sortie :

   ```
   "ECS_DATADIR=/data",
   ```

> **Important**
>
> Si la commande précédente ne renvoie pas la variable d'environnement `ECS_DATADIR`, vous devez arrêter toutes les tâches en cours d'exécution sur cette instance de conteneur avant d'arrêter l'agent. Les agents les plus récents avec la variable d'environnement `ECS_DATADIR` enregistrent leur état et peuvent être arrêtés et démarrés sans problèmes tandis que des tâches sont en cours d'exécution. Pour de plus amples informations, veuillez consulter Mise à jour de l'agent de conteneur Amazon ECS.

4. Arrêtez le service `ecs` :

```
sudo stop ecs
```

Sortie :

```
ecs stop/waiting
```

5. Redémarrez le service `ecs`.

   - Pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS :

     ```
     sudo systemctl restart ecs
     ```

   - Pour l'AMI Amazon Linux optimisée pour Amazon ECS :

     ```
     sudo stop ecs && sudo start ecs
     ```

6. (Facultatif) Vous pouvez vérifier que l'agent est en cours d'exécution et consulter des informations sur votre nouvelle instance de conteneur en interrogeant l'opération API d'introspection d'agent. Pour de plus amples informations, veuillez consulter the section called “Introspection des conteneurs”.

```
curl http://localhost:51678/v1/metadata
```

Nettoyage automatique des tâches et des images Amazon ECS

Chaque fois qu'une tâche est placée sur une instance de conteneur, l'agent de conteneur Amazon ECS vérifie si les images référencées dans la tâche sont les images les plus récentes de la balise spécifiée dans le référentiel. Si ce n'est pas le cas, le comportement par défaut permet à l'agent d'extraire les images de leurs référentiels respectifs. Si vous modifiez fréquemment les images dans vos tâches et services, votre stockage d'instance de conteneur peut se remplir rapidement avec des images Docker que vous n'utilisez plus et que vous n'utiliserez probablement jamais plus. Par exemple, vous utilisez peut-être un pipeline pour l'intégration et le déploiement continus (CI/CD).

> **Note**
>
> Il est possible de personnaliser le comportement d'extraction d'image de l'agent Amazon ECS à l'aide du paramètre `ECS_IMAGE_PULL_BEHAVIOR`. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

De même, les conteneurs appartenant à des tâches arrêtées peuvent également consommer du stockage d'instance de conteneur avec des informations de journal, des volumes de données et d'autres artefacts. Ces artefacts sont utiles pour le débogage des conteneurs qui se sont arrêtés de manière inattendue, mais la plupart de ce stockage peut être libéré en toute sécurité après une période donnée.

Par défaut, l'agent de conteneur Amazon ECS élimine automatiquement les tâches arrêtées et les images Docker qui ne sont pas utilisées par des tâches de vos instances de conteneur.

> **Note**
>
> La fonction de nettoyage automatique d'image nécessite au moins la version 1.13.0 de l'agent de conteneur Amazon ECS. Pour mettre à jour votre agent avec la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Les variables de configuration d'agent suivantes sont disponibles pour ajuster votre expérience de tâches automatisées et de nettoyage d'image. Pour plus d'informations sur la façon de définir ces variables sur vos instances de conteneur, consultez Configuration de l'agent de conteneur Amazon ECS.

`ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`

Temps d'attente par défaut pour supprimer des conteneurs pour une tâche arrêtée. Si la valeur est définie sur moins d'une seconde, elle est ignorée. Par défaut, ce paramètre est défini sur trois heures, mais vous pouvez réduire cette période à une seconde, si votre application le nécessite.

> **Note**
>
> Le processus de nettoyage d'image ne peut pas supprimer une image tant qu'un conteneur y fait référence. Une fois les conteneurs supprimés, toutes les images non référencées peuvent être nettoyées en fonction des paramètres de configuration du nettoyage d'image.

`ECS_DISABLE_IMAGE_CLEANUP`

Si vous définissez cette variable sur `true`, le nettoyage automatique d'image est désactivé sur votre instance de conteneur et aucune image n'est supprimée automatiquement.

`ECS_IMAGE_CLEANUP_INTERVAL`

Cette variable spécifie à quelle fréquence le processus de nettoyage d'image automatique recherche des images à supprimer. La valeur par défaut est toutes les 30 minutes, mais vous pouvez réduire ce délai à 10 minutes pour supprimer les images plus fréquemment.

`ECS_IMAGE_MINIMUM_CLEANUP_AGE`

Cette variable spécifie le délai minimal entre le moment où une image a été extraite et celui où elle peut être supprimée. Cela permet d'empêcher le nettoyage d'images tout juste extraites. La valeur par défaut est 1 heure.

`ECS_NUM_IMAGES_DELETE_PER_CYCLE`

Cette variable spécifie le nombre d'images pouvant être supprimées en un seul cycle de nettoyage. La valeur par défaut est de 5 et la valeur minimale est de 1.

Lorsque l'agent de conteneur Amazon ECS est en cours d'exécution et que le nettoyage automatique d'image n'est pas désactivé, l'agent recherche des images Docker qui ne sont pas référencées par des conteneurs en cours d'exécution ou arrêtés à une fréquence déterminée par la variable `ECS_IMAGE_CLEANUP_INTERVAL`. Si des images inutilisées sont trouvées et qu'elles sont antérieures au délai de nettoyage minimal spécifié par la variable

`ECS_IMAGE_MINIMUM_CLEANUP_AGE`, l'agent supprime le nombre maximal d'images spécifiées avec la variable `ECS_NUM_IMAGES_DELETE_PER_CYCLE`. Les images référencées le moins récemment sont supprimées en premier. Une fois les images supprimées, l'agent attend jusqu'au prochain intervalle et répète le processus.

Planifiez vos conteneurs sur Amazon ECS

Amazon Elastic Container Service (Amazon ECS) est un système de simultanéité optimiste à états partagés qui fournit des fonctionnalités de planification flexibles pour vos charges de travail conteneurisées. Les planificateurs Amazon ECS utilisent les informations d'état de cluster fournies par l'API Amazon ECS pour prendre des décisions de placement appropriées.

Amazon ECS fournit un planificateur de service pour les tâches et les applications de longue durée. Il permet également d'exécuter des tâches autonomes ou des tâches planifiées pour des tâches par lots ou des tâches à exécution unique. Vous pouvez spécifier les stratégies et les contraintes de placement des tâches pour exécuter les tâches qui répondent le mieux à vos besoins. Par exemple, vous pouvez spécifier si les tâches s'exécutent sur plusieurs zones de disponibilité ou dans une seule zone de disponibilité. Vous pouvez intégrer les tâches avec vos propres planificateurs personnalisés ou tiers.

Option	Utilisation	En savoir plus
Service	Le planificateur de services convient aux services et applications apatrides de longue durée. Le planificateur de service peut aussi s'assurer que les tâches sont enregistr ées sur un équilibreur de charge Elastic Load Balancing . Vous pouvez mettre à jour vos services gérés par le planificateur de service. Cela peut inclure le déploieme nt d'une nouvelle définitio n de tâche ou la modificat ion du nombre de tâches souhaitées en cours d'exécuti on. Par défaut, le planifica teur de service répartit les tâches entre plusieurs zones	Services Amazon ECS

Option	Utilisation	En savoir plus
	de disponibilité. Cependant , vous pouvez utiliser des stratégies et des contraintes de placement des tâches afin de personnaliser la façon dont les tâches sont placées.	
Tâche autonome	Une tâche autonome convient aux processus tels que les tâches par lots qui exécutent un travail puis s'arrêtent. Par exemple, un processus peut appeler `RunTask` lorsque le travail passe en file d'attente . La tâche extrait le travail de la file d'attente, effectue le travail, puis se termine. Avec `RunTask`, vous pouvez permettre à la stratégie de placement des tâches par défaut de distribuer les tâches de façon aléatoire sur le cluster. Cela limite les risques qu'une seule instance reçoive un nombre de tâches disproportionné.	Tâches autonomes Amazon ECS

Option	Utilisation	En savoir plus
Tâches planifiées	Une tâche planifiée convient lorsque vous avez des tâches à exécuter à des intervalles définis dans votre cluster. Vous pouvez utiliser le EventBridge planificateur pour créer un calendrier. Vous pouvez exécuter des tâches pour une opération de sauvegarde ou une analyse des journaux. Le calendrier du EventBridge planificateur que vous créez peut exécuter une ou plusieurs tâches dans votre cluster à des moments précis. Votre événement planifié peut être défini sur un intervalle spécifique (exécuté toutes les *N* minutes, toutes les heures ou tous les jours). Sinon, pour une planification plus compliquée, vous pouvez utiliser une expression `cron`.	Utilisation d'Amazon EventBridge Scheduler pour planifier des tâches Amazon ECS

Options de calcul

Avec Amazon ECS, vous pouvez spécifier l'infrastructure sur laquelle vos tâches ou services s'exécutent. Vous pouvez utiliser une stratégie de fournisseur de capacité ou un type de lancement.

Pour Fargate, les fournisseurs de capacité sont Fargate et Fargate Spot. En EC2 effet, le fournisseur de capacité est le groupe Auto Scaling avec les instances de conteneur enregistrées.

La stratégie des fournisseurs de capacité répartit vos tâches entre les fournisseurs de capacité associés à votre cluster.

Seuls les fournisseurs de capacité déjà associés à un cluster et disposant d'un statut `ACTIVE` ou `UPDATING` peuvent être utilisés dans une stratégie de fournisseur de capacité. Vous pouvez associer un fournisseur de capacité à un cluster lorsque vous créez un cluster.

Dans une stratégie de fournisseur de capacité, la valeur de base facultative indique le nombre minimum de tâches qui s'exécutent sur un fournisseur de capacité spécifié. Une base ne peut être définie que pour un seul fournisseur de capacité dans une stratégie de fournisseur de capacité.

La valeur de poids détermine le pourcentage relatif du nombre total de tâches lancées qui utilisent le fournisseur de capacité spécifié. Prenez l'exemple de code suivant. Vous avez une stratégie qui contient deux fournisseurs de capacité et que les deux ont un poids de `1`. Lorsque le pourcentage de base est atteint, les tâches sont réparties équitablement entre les deux fournisseurs de capacité. Dans la même logique, supposons que vous spécifiez un poids de `1` pour capacityProviderA et un poids de `4` pour capacityProviderB. Ensuite, pour chaque tâche exécutée avec capacityProviderA, il y a quatre tâches qui utilisent capacityProviderB.

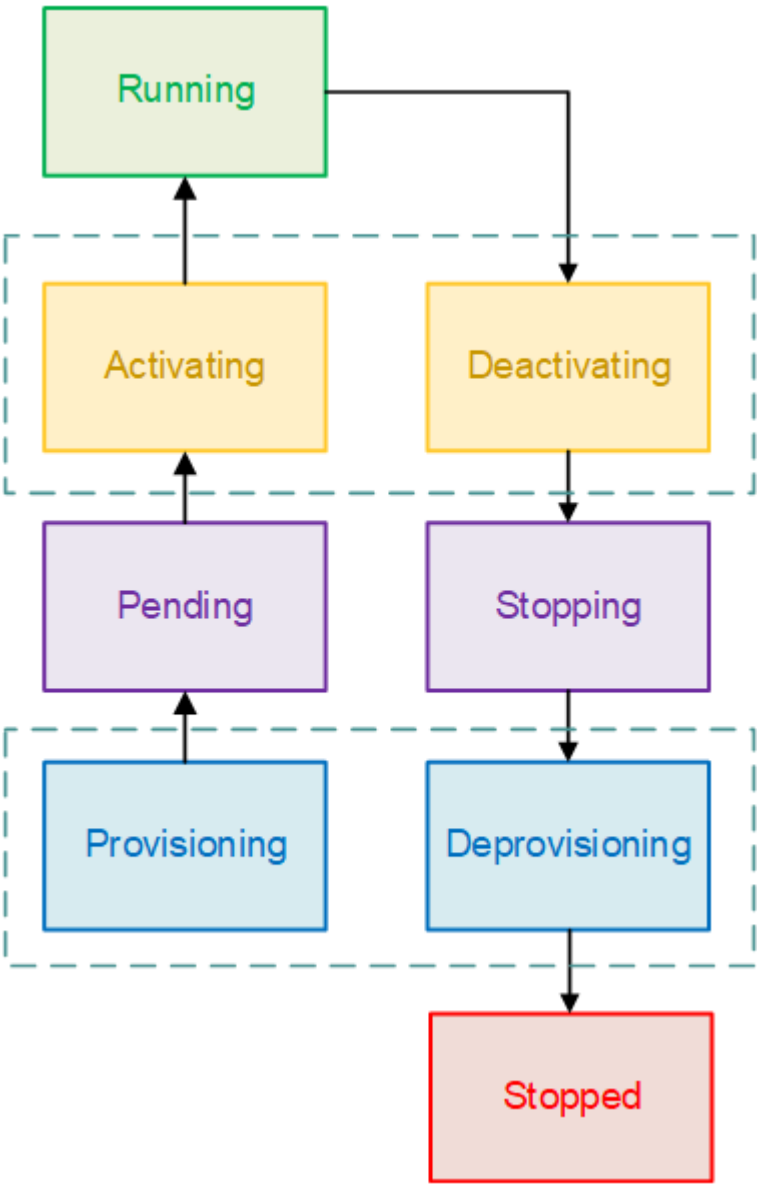
Le type de lancement lance vos tâches directement sur Fargate ou sur les instances EC2 Amazon que vous avez enregistrées manuellement dans vos clusters.

Cycle de vie des tâches Amazon ECS

Lorsqu'une tâche est lancée, soit manuellement ou dans le cadre d'un service, elle peut passer par différents états avant de se terminer d'elle-même ou d'être arrêtée manuellement. Certaines tâches sont destinées à s'exécuter en tant que tâches de traitement par lots qui passent naturellement de `PENDING` à `RUNNING` à `STOPPED`. D'autres tâches, qui peuvent être lancées dans le cadre d'un service, sont censées continuer à s'exécuter indéfiniment, ou à évoluer en fonction des besoins.

Lorsque des modifications de l'état d'une tâche sont demandées, par exemple l'arrêt d'une tâche ou la mise à jour du nombre souhaité pour un service pour le mettre à l'échelle, l'agent de conteneur Amazon ECS considère ces modifications comme le dernier état connu (`lastStatus`) de la tâche et état souhaité (`desiredStatus`) de la tâche. Le dernier état connu et l'état souhaité d'une tâche peuvent être vus dans la console ou en décrivant la tâche avec l'API ou l' AWS CLI.

Le diagramme ci-dessous illustre le flux du cycle de vie des tâches.



États de cycle de vie

Voici des descriptions de chacun des états du cycle de vie de tâche.

PROVISIONING

Amazon ECS doit effectuer des étapes supplémentaires avant que la tâche soit lancée. Par exemple, pour les tâches qui utilisent le mode de réseau `awsvpc`, l'interface réseau Elastic doit être mise en service.

PENDING

Il s'agit d'un état de transition où Amazon ECS attend l'agent de conteneur pour prendre les mesures nécessaires. Une tâche reste en attente jusqu'à ce que des ressources soient disponibles pour la tâche.

ACTIVATING

Il s'agit d'un état de transition lors duquel Amazon ECS doit effectuer des étapes supplémentaires une fois que la tâche est lancée, mais avant que la tâche passe à l'état `RUNNING`. Il s'agit de l'état dans lequel Amazon ECS extrait les images des conteneurs, crée les conteneurs, configure le réseau des tâches, enregistre les groupes cibles des équilibreurs de charge et configure la découverte des services.

RUNNING

La tâche est en cours d'exécution.

DEACTIVATING

Il s'agit d'un état de transition lors duquel Amazon ECS doit effectuer des étapes supplémentaires avant que la tâche soit arrêtée. Par exemple, pour les tâches faisant partie d'un service configuré pour utiliser les groupes cibles d'Elastic Load Balancings, la désinscription du groupe cible a lieu pendant cet état.

STOPPING

Il s'agit d'un état de transition où Amazon ECS attend l'agent de conteneur pour prendre les mesures nécessaires.

Pour les conteneurs Linux, l'agent de conteneur envoie le `SIGTERM` signal pour indiquer que l'application doit terminer et s'arrêter, puis envoie un signal `SIGKILL` après avoir attendu la `StopTimeout` durée définie dans la définition de la tâche.

DEPROVISIONING

Amazon ECS doit effectuer des étapes supplémentaires une fois que la tâche est arrêtée, mais avant que la tâche passe à l'état `STOPPED`. Par exemple, pour les tâches qui utilisent le mode de réseau `awsvpc`, l'interface réseau Elastic doit être détachée et supprimée.

STOPPED

La tâche a été arrêtée avec succès.

Si votre tâche s'est arrêtée en raison d'une erreur, consultez L'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

SUPPRIMÉ

Il s'agit d'un état de transition lorsqu'une tâche s'arrête. Cet état ne s'affiche pas sur la console, mais s'affiche sur les `describe-tasks`.

Comment Amazon ECS place les tâches sur les instances de conteneur

Vous pouvez utiliser le placement des tâches pour configurer Amazon ECS afin de placer vos tâches sur des instances de conteneur répondant à certains critères, par exemple une zone de disponibilité ou un type d'instance.

Les composants de placement des tâches sont les suivants :

- Stratégie de placement des tâches - Algorithme de sélection des instances de conteneur pour le placement des tâches ou des tâches à terminer. Par exemple, Amazon ECS peut sélectionner des instances de conteneur de manière aléatoire, ou il peut sélectionner des instances de conteneur de manière à ce que les tâches soient réparties uniformément sur un groupe d'instances.
- Groupe de tâches : groupe de tâches connexes, par exemple des tâches de base de données.
- Contrainte de placement des tâches : il s'agit de règles qui doivent être respectées pour placer une tâche sur une instance de conteneur. Si la contrainte n'est pas respectée, la tâche n'est pas placée et reste dans son `PENDING` état. Par exemple, vous pouvez utiliser une contrainte pour placer des tâches uniquement sur un type d'instance spécifique.

Amazon ECS utilise différents algorithmes pour les types de lancement.

EC2 type de lancement

Pour les tâches qui utilisent le type de EC2 lancement, Amazon ECS doit déterminer où placer la tâche en fonction des exigences spécifiées dans la définition de la tâche, telles que le processeur et la mémoire. De la même manière, lorsque vous réduisez le nombre de tâches, Amazon ECS doit déterminer quelles tâches doivent être résiliées. Vous pouvez appliquer des stratégies et des contraintes de placement des tâches afin de personnaliser la façon dont Amazon ECS place et résilie les tâches.

Les stratégies de placement des tâches par défaut varient selon que vous les exécutez manuellement (tâches autonomes) ou au sein d'un service. Pour les tâches exécutées dans le cadre d'un service Amazon ECS, la stratégie de placement des tâches est `spread`, en utilisant `attribute:ecs.availability-zone`. Il n'existe pas de contrainte de placement des tâches par défaut pour les tâches ne figurant pas dans les services. Pour de plus amples informations, veuillez consulter Planifiez vos conteneurs sur Amazon ECS.

> **Note**
>
> Les stratégies de placement des tâches ont une obligation de moyens, mais pas de résultat. Amazon ECS tente toujours de placer les tâches, même lorsque l'option de placement optimale n'est pas disponible. Néanmoins, les contraintes de placement des tâches constituent une obligation et peuvent empêcher le placement des tâches.

Vous pouvez utiliser simultanément des stratégies et des contraintes de placement des tâches. Par exemple, vous pouvez utiliser une stratégie et une contrainte de placement des tâches de façon à répartir les tâches entre différentes zones de disponibilité et les regrouper par bin packing en fonction de la mémoire dans chaque zone de disponibilité, mais uniquement pour les instances G2.

Lorsqu'Amazon ECS place des tâches, il a recours au processus suivant afin de sélectionner les instances de conteneur :

1. Identifiez les instances de conteneur qui répondent aux exigences en termes de processeur, de processeur graphique, de mémoire et de port dans la définition de la tâche.
2. Identifiez les instances de conteneur qui répondent aux contraintes de placement des tâches.
3. Identifiez les instances de conteneur qui répondent aux stratégies de placement des tâches.
4. Sélectionnez les instances de conteneur pour le placement des tâches.

Type de lancement Fargate

Les stratégies et contraintes de placement de tâche ne sont pas prises en charge pour les tâches utilisant le type de lancement Fargate. Fargate fera de son mieux pour répartir les tâches entre les zones de disponibilité accessibles. Si le fournisseur de capacité inclut à la fois Fargate et Fargate Spot, le comportement de répartition est indépendant pour chaque fournisseur de capacité.

Utilisez des stratégies pour définir le placement des tâches Amazon ECS

Pour les tâches qui utilisent le type de EC2 lancement, Amazon ECS doit déterminer où placer la tâche en fonction des exigences spécifiées dans la définition de la tâche, telles que le processeur et la mémoire. De la même manière, lorsque vous réduisez le nombre de tâches, Amazon ECS doit déterminer quelles tâches doivent être résiliées. Vous pouvez appliquer des stratégies et des contraintes de placement des tâches afin de personnaliser la façon dont Amazon ECS place et résilie les tâches.

Les stratégies de placement des tâches par défaut varient selon que vous les exécutez manuellement (tâches autonomes) ou au sein d'un service. Pour les tâches exécutées dans le cadre d'un service Amazon ECS, la stratégie de placement des tâches est `spread`, en utilisant `attribute:ecs.availability-zone`. Il n'existe pas de contrainte de placement des tâches par défaut pour les tâches ne figurant pas dans les services. Pour de plus amples informations, veuillez consulter Planifiez vos conteneurs sur Amazon ECS.

> **Note**
>
> Les stratégies de placement des tâches ont une obligation de moyens, mais pas de résultat. Amazon ECS tente toujours de placer les tâches, même lorsque l'option de placement optimale n'est pas disponible. Néanmoins, les contraintes de placement des tâches constituent une obligation et peuvent empêcher le placement des tâches.

Vous pouvez utiliser simultanément des stratégies et des contraintes de placement des tâches. Par exemple, vous pouvez utiliser une stratégie et une contrainte de placement des tâches de façon à répartir les tâches entre différentes zones de disponibilité et les regrouper par bin packing en fonction de la mémoire dans chaque zone de disponibilité, mais uniquement pour les instances G2.

Lorsqu'Amazon ECS place des tâches, il a recours au processus suivant afin de sélectionner les instances de conteneur :

1. Identifiez les instances de conteneur qui répondent aux exigences en termes de processeur, de processeur graphique, de mémoire et de port dans la définition de la tâche.
2. Identifiez les instances de conteneur qui répondent aux contraintes de placement des tâches.
3. Identifiez les instances de conteneur qui répondent aux stratégies de placement des tâches.
4. Sélectionnez les instances de conteneur pour le placement des tâches.

Vous spécifiez les stratégies de placement des tâches dans la définition du service ou dans la définition des tâches à l'aide du `placementStrategy` paramètre.

```
"placementStrategy": [
    {
        "field": "The field to apply the placement strategy against",
        "type": "The placement strategy to use"
    }
]
```

Vous pouvez définir les stratégies lorsque vous exécutez une tâche (RunTask), créez un nouveau service (CreateService) ou mettez à jour un service existant (UpdateService).

Le tableau suivant décrit les types et les champs disponibles.

type	Valeurs de champ valides	
`binpack` Les tâches sont placées sur les instances de conteneur afin de laisser le moins de CPU ou de mémoire inutilisé es. Cette stratégie minimise le nombre d'instances de conteneur utilisées. Lorsque cette stratégie est utilisée et qu'une action de mise à l'échelle horizontale est entreprise, Amazon ECS résilie les tâches. Elle effectue cette opération en fonction de la quantité de ressource s laissée sur l'instance de conteneur après la résiliati on de la tâche. L'instance de conteneur qui a le plus de ressources disponibles après	• cpu • memory	

type	Valeurs de champ valides	
la résiliation de la tâche voit cette tâche résiliée.		
`random` Les tâches sont placées au hasard.	Non utilisé	
`spread` Les tâches sont placées uniformément en fonction de la valeur spécifiée. Les tâches de service sont réparties en fonction des tâches du service concerné. Les tâches autonomes sont réparties en fonction des tâches du même groupe de tâches. Pour de plus amples informations sur les groupes de tâches, consultez Tâches Amazon ECS liées au groupe . Lorsque la stratégie `spread` est utilisée et qu'une action de mise à l'échelle horizontale est entreprise, Amazon ECS sélectionne les tâches à résilier qui maintiennent un équilibre entre les zones de disponibilité. Au sein d'une zone de disponibilité, les tâches sont sélectionnées au hasard.	• `instanceId` (ou`host`, ce qui a le même effet) • toute plate-forme ou tout attribut personnalisé appliqué à une instance de conteneur, tel que `attribute:ecs.availability-zone`	

Les stratégies de placement des tâches peuvent également être mises à jour pour les services existants. Pour de plus amples informations, veuillez consulter Comment Amazon ECS place les tâches sur les instances de conteneur.

Vous pouvez créer une stratégie de placement des tâches qui utilise plusieurs stratégies en créant des tableaux de stratégies dans l'ordre d'exécution souhaité. Par exemple, si vous souhaitez répartir les tâches entre les zones de disponibilité, puis les regrouper en fonction de la mémoire au sein de chaque zone de disponibilité, spécifiez la stratégie de zone de disponibilité suivie de la stratégie de mémoire. Pour consulter des exemples de stratégies, voir Exemples de stratégies de placement de tâches Amazon ECS.

Exemples de stratégies de placement de tâches Amazon ECS

Vous pouvez définir des stratégies de placement des tâches à l'aide des actions suivantes : CreateServiceUpdateService, et RunTask.

Exemples

- Répartition des tâches de manière uniforme sur plusieurs zones de disponibilité
- Répartition des tâches de manière uniforme sur toutes les instances
- Regroupement des tâches en fonction de la mémoire
- Placement des tâches de façon aléatoire
- Répartition des tâches de manière uniforme entre les zones de disponibilité, puis répartition de manière uniforme entre les instances de chaque zone de disponibilité
- Répartition des tâches de manière uniforme entre les zones de disponibilité, puis regroupement des tâches en fonction de la mémoire de chaque zone de disponibilité
- Répartition des tâches de manière uniforme entre les instances, puis regroupement des tâches en fonction de la mémoire

Répartition des tâches de manière uniforme sur plusieurs zones de disponibilité

La stratégie suivante répartit les tâches de façon uniforme dans les zones de disponibilité.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    }
```

```
]
```

Répartition des tâches de manière uniforme sur toutes les instances

La stratégie suivante répartit les tâches de façon uniforme entre toutes les instances.

```
"placementStrategy": [
    {
        "field": "instanceId",
        "type": "spread"
    }
]
```

Regroupement des tâches en fonction de la mémoire

La stratégie suivante regroupe les tâches par bin packing en fonction de la mémoire.

```
"placementStrategy": [
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Placement des tâches de façon aléatoire

La stratégie suivante place les tâches de façon aléatoire.

```
"placementStrategy": [
    {
        "type": "random"
    }
]
```

Répartition des tâches de manière uniforme entre les zones de disponibilité, puis répartition de manière uniforme entre les instances de chaque zone de disponibilité

La stratégie suivante permet de répartir les tâches de façon égale entre les zones de disponibilité, puis de les répartir de façon égale entre les instances de chaque zone de disponibilité.

```
"placementStrategy": [
```

```
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "instanceId",
        "type": "spread"
    }
]
```

Répartition des tâches de manière uniforme entre les zones de disponibilité, puis regroupement des tâches en fonction de la mémoire de chaque zone de disponibilité

La stratégie suivante permet de répartir les tâches de façon égale entre les zones de disponibilité, puis de répartir les tâches par bin packing en fonction de la mémoire de chaque zone de disponibilité.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Répartition des tâches de manière uniforme entre les instances, puis regroupement des tâches en fonction de la mémoire

La stratégie suivante répartit les tâches de manière uniforme entre toutes les instances, puis regroupe les tâches en fonction de la mémoire au sein de chaque instance.

```
"placementStrategy": [
    {
        "field": "instanceId",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
```

```
]
```

Tâches Amazon ECS liées au groupe

Vous pouvez identifier un ensemble de tâches connexes et les placer dans un groupe de tâches. Toutes les tâches ayant le même nom de groupe de tâches sont considérées comme un ensemble lorsque la stratégie de placement des tâches `spread` est utilisée. Par exemple, supposons que vous exécutiez différentes applications dans un cluster, par exemple des bases de données et des serveurs web. Afin de veiller à ce que les bases de données soient réparties de façon équilibrée dans les zones de disponibilité, ajoutez-les à un groupe de tâches nommé `databases`, puis utilisez la stratégie de placement des tâches `spread`. Pour de plus amples informations, veuillez consulter Utilisez des stratégies pour définir le placement des tâches Amazon ECS.

Les groupes de tâches peuvent également être utilisés comme contrainte de placement des tâches. Lorsque vous spécifiez un groupe de tâches dans la contrainte `memberOf`, les tâches sont uniquement envoyées aux instances de conteneur qui se trouvent dans le groupe de tâches spécifié. Pour obtenir un exemple, consultez Exemple de contraintes de placement de tâches Amazon ECS.

Par défaut, les tâches autonomes utilisent le nom de famille de définition de tâche (par exemple, `family:my-task-definition`) comme nom de groupe de tâches si aucun nom de groupe de tâches personnalisé n'est spécifié. Les tâches lancées dans le cadre d'un service utilisent le nom du service comme nom de groupe de tâches et ne peuvent pas être modifiées.

Les exigences suivantes s'appliquent au groupe de travail.

- Un nom de groupe de tâches doit être composé de 255 caractères maximum.
- Chaque tâche peut faire partie d'un seul groupe.
- Après avoir lancé une tâche, vous ne pouvez pas modifier le groupe de tâches auquel elle appartient.

Définissez les instances de conteneur qu'Amazon ECS utilise pour les tâches

Une contrainte de placement de tâche est une règle concernant une instance de conteneur qu'Amazon ECS utilise pour déterminer si la tâche est autorisée à s'exécuter sur l'instance. Au moins une instance de conteneur doit correspondre à la contrainte. Si aucune instance ne correspond à la contrainte, la tâche reste à l'état `PENDING`. Lorsque vous créez un nouveau service ou que vous

mettez à jour un service existant, vous pouvez définir des contraintes de placement des tâches pour les tâches du service.

Vous pouvez spécifier les contraintes de placement des tâches dans la définition du service, la définition de la tâche ou la tâche à l'aide du `placementConstraint` paramètre.

```
"placementConstraints": [
    {
        "expression": "The expression that defines the task placement constraints",
        "type": "The placement constraint type to use"
    }
]
```

Le tableau suivant décrit comment utiliser les paramètres.

Constraint type (Type de contrainte)	Peut être spécifié quand	
`distinctInstance` Placez chaque tâche active sur une instance de conteneur différente. Amazon ECS examine le statut souhaité des tâches pour le placement des tâches. Par exemple, si le statut souhaité de la tâche existante est`STOPPED`, (mais pas le dernier statut), une nouvelle tâche entrante peut être placée sur la même instance malgré la contraint e de `distinctInstance` placement. Par conséquen t, vous pouvez voir deux tâches dont le dernier statut	• Exécution d'une tâche RunTask • Création d'un nouveau service CreateService,	

Constraint type (Type de contrainte)	Peut être spécifié quand	
est `RUNNING` sur la même instance. **Important** Nous recommandons aux clients qui recherchent une isolation solide pour leurs tâches d'utiliser Fargate. Fargate exécute chaque tâche dans un environnement de virtualisation matérielle. Cela garantit que ces charges de travail conteneurisées ne partagent pas les interfaces réseau, le stockage éphémère Fargate, le processeur ou la mémoire avec d'autres tâches. Pour plus d'informations, consultez la section Présentation de la sécurité de AWS Fargate.		

Constraint type (Type de contrainte)	Peut être spécifié quand	
`memberOf` Place les tâches sur des instances de conteneur qui correspondent à une expression.	• Exécution d'une tâche RunTask • Création d'un nouveau service CreateService, • Création d'une nouvelle définition de tâche RegisterTaskDefinition • Création d'une nouvelle révision d'une définition de tâche RegisterTaskDefinition • Mettre à jour un service UpdateService	

Lorsque vous utilisez le type de `memberOf` contrainte, vous pouvez créer une expression à l'aide du langage de requête de cluster qui définit les instances de conteneur dans lesquelles Amazon ECS peut placer des tâches. L'expression vous permet de regrouper vos instances de conteneur par attributs. L'expression entre dans le `expression` paramètre `deplacementConstraint`.

Attributs de l'instance de conteneur Amazon ECS

Vous pouvez ajouter à vos instances de conteneur des métadonnées personnalisées connues sous le nom d'attributs. Chaque attribut a un nom et une valeur de chaîne facultative. Vous pouvez utiliser les attributs intégrés fournis par Amazon ECS ou définir des attributs personnalisés.

Les sections suivantes contiennent des exemples d'attributs intégrés, facultatifs et personnalisés.

Attributs intégrés

Amazon ECS applique automatiquement les attributs suivants à vos instances de conteneur.

`ecs.ami-id`

ID de l’AMI utilisée pour lancer l’instance. Cet attribut peut par exemple avoir la valeur `ami-1234abcd`.

`ecs.availability-zone`

Zone de disponibilité de l'instance. Cet attribut peut par exemple avoir la valeur `us-east-1a`.

`ecs.instance-type`

Type de l'instance. Cet attribut peut par exemple avoir la valeur `g2.2xlarge`.

`ecs.os-type`

Système d'exploitation de l'instance. Les valeurs possibles pour cet attribut sont `linux` et `windows`.

`ecs.os-family`

Version du système d'exploitation de l'instance.

Pour les instances Linux, la valeur valide est `LINUX`. Pour les instances Windows, ECS définit la valeur au format `WINDOWS_SERVER_<OS_Release>_<FULL or CORE>`. Les valeurs valides sont `WINDOWS_SERVER_2022_FULL`, `WINDOWS_SERVER_2022_CORE`, `WINDOWS_SERVER_20H2_CORE`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE` et `WINDOWS_SERVER_2016_FULL`.

Cela est important pour les conteneurs Windows et Windows containers on AWS Fargate parce que la version du système d'exploitation de chaque conteneur Windows doit correspondre à celle de l'hôte. Si la version Windows de l'image du conteneur est différente de celle de l'hôte, le conteneur ne démarre pas. Pour plus d'informations, consultez Compatibilité avec la version du conteneur Windows sur le site web de documentation Microsoft.

Si votre cluster exécute plusieurs versions de Windows, vous pouvez vous assurer qu'une tâche est placée sur une EC2 instance exécutée sur la même version en utilisant la contrainte de placement :`memberOf(attribute:ecs.os-family == WINDOWS_SERVER_<OS_Release>_<FULL or CORE>)`. Pour de plus amples informations, veuillez consulter the section called “Récupération des métadonnées de l'AMI Windows optimisées pour Amazon ECS”.

`ecs.cpu-architecture`

Architecture du processeur de l'instance. Cet attribut peut par exemple avoir la valeur `x86_64` ou `arm64`.

`ecs.vpc-id`

VPC dans lequel l'instance a été lancée. Cet attribut peut par exemple avoir la valeur `vpc-1234abcd`.

`ecs.subnet-id`

Sous-réseau utilisé par l'instance. Cet attribut peut par exemple avoir la valeur `subnet-1234abcd`.

Attributs facultatifs

Amazon ECS peut ajouter les attributs suivants à vos instances de conteneur.

`ecs.awsvpc-trunk-id`

Si cet attribut existe, l'instance dispose d'une interface réseau de jonction. Pour de plus amples informations, veuillez consulter Augmenter les interfaces réseau des instances de conteneur Linux Amazon ECS.

`ecs.outpost-arn`

Si cet attribut existe, il contient l'Amazon Resource Name (ARN) de l'Outpost. Pour de plus amples informations, veuillez consulter the section called “Amazon Elastic Container Service sur AWS Outposts”.

`ecs.capability.external`

Si cet attribut existe, l'instance est identifiée en tant qu'instance externe. Pour de plus amples informations, veuillez consulter Clusters Amazon ECS pour le type de lancement externe.

Attributs personnalisés

Vous pouvez appliquer des attributs personnalisés à vos instances de conteneur. Par exemple, vous pouvez définir un attribut avec le nom « stack » et la valeur « prod ».

Lorsque vous spécifiez des attributs personnalisés, vous devez prendre en compte les éléments suivants.

- Le `name` doit comprendre entre 1 et 128 caractères et peut contenir des lettres (majuscules et minuscules), des chiffres, des tirets, des traits de soulignement, des barres obliques et des points.
- Le `value` doit comprendre entre 1 et 128 caractères et peut contenir des lettres (majuscules et minuscules), des chiffres, des tirets, des traits de soulignement, des points, des arrobases, des barres obliques, des deux-points et des espaces. La valeur ne peut pas contenir d'espaces de début ou de fin.

Création d'expressions pour définir des instances de conteneur pour les tâches Amazon ECS

Les requêtes de cluster sont des expressions qui vous permettent de regrouper des objets. Par exemple, vous pouvez regrouper des instances de conteneur par attributs, notamment par zone de disponibilité, type d'instance ou métadonnées personnalisées. Pour de plus amples informations, veuillez consulter Attributs de l'instance de conteneur Amazon ECS.

Une fois que vous avez défini un groupe d'instances de conteneur, vous pouvez personnaliser Amazon ECS de façon à placer les instances de conteneur en fonction du groupe. Pour plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS et Création d'un déploiement de mises à jour continues Amazon ECS. Vous pouvez également appliquer un filtre de groupe lorsque vous répertoriez les instances de conteneur.

Syntaxe des expressions

Les expressions ont la syntaxe suivante :

```
subject operator [argument]
```

Sujet

Attribut ou champ à évaluer.

`agentConnected`

Sélectionnez les instances de conteneur par leur état de connexion de l'agent de conteneur Amazon ECS. Vous pouvez utiliser ce filtre pour rechercher des instances avec des agents de conteneur déconnectés.

Opérateurs valides : equals (==), not_equals (!=), in, not_in (!in), matches (=~), not_matches (!~)

`agentVersion`

Sélectionnez les instances de conteneur par la version de l'agent de conteneur Amazon ECS. Vous pouvez utiliser ce filtre pour trouver des instances qui exécutent des versions obsolètes de l'agent de conteneur Amazon ECS.

Opérateurs valides : equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

`attribute:attribute-name`

Sélectionnez les instances de conteneur par attribut. Pour de plus amples informations, veuillez consulter Attributs de l'instance de conteneur Amazon ECS.

`ec2InstanceId`

Sélectionnez les instances de conteneur en fonction de leur ID d' EC2 instance Amazon.

Opérateurs valides : equals (==), not_equals (!=), in, not_in (!in), matches (=~), not_matches (!~)

`registeredAt`

Sélectionnez les instances de conteneur par leur date d'enregistrement d'instance de conteneur. Vous pouvez utiliser ce filtre pour trouver de nouvelles instances enregistrées ou des instances très anciennes.

Opérateurs valides : equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

Formats de date valides : 2018-06-18T22:28:28+00:00, 2018-06-18T22:28:28Z, 2018-06-18T22:28:28, 2018-06-18

`runningTasksCount`

Sélectionnez les instances de conteneur par nombre de tâches en cours d'exécution. Vous pouvez utiliser ce filtre pour trouver des instances vides ou presque vides (peu de tâches exécutées sur elles).

Opérateurs valides : equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

`task:group`

Sélectionnez les instances de conteneur par groupe de tâches. Pour de plus amples informations, veuillez consulter Tâches Amazon ECS liées au groupe .

Opérateur

Opérateur de comparaison. Les opérateurs suivants sont pris en charge.

Opérateur	Description
==, equals	Égalité de chaînes

Opérateur	Description
!=, not_equals	Inégalité de chaînes
>, greater_than	Supérieur à
>=, greater_than_equal	Supérieur ou égal à
<, less_than	Inférieur à
<=, less_than_equal	Inférieur ou égal à
exists	Le sujet existe
!exists, not_exists	Le sujet n'existe pas
dans	Valeur présente dans la liste d'arguments
!in, not_in	Valeur ne se trouvant pas dans la liste d'arguments
=~, matches	Correspondance de modèle
!~, not_matches	Non-correspondance de modèle

> **Note**
>
> Une expression ne peut pas contenir de parenthèses. En revanche, vous pouvez utiliser des parenthèses pour spécifier la priorité dans des expressions composées.

Argument

Pour de nombreux opérateurs, l'argument est une valeur littérale.

Les opérateurs `in` et `not_in` attendent une liste d'arguments comme argument. Vous spécifiez une liste d'arguments comme suit :

```
[argument1, argument2, ..., argumentN]
```

Les opérateurs matches et not_matches attendent un argument respectant la syntaxe de l'expression régulière Java. Pour en savoir plus, consultez java.util.regex.Pattern.

Expressions composées

Vous pouvez combiner des expressions à l'aide des opérateurs booléens suivants :

- &&, and
- ||, or
- !, not

Vous pouvez spécifier la priorité à l'aide de parenthèses :

```
(expression1 or expression2) and expression3
```

Exemples d'expressions

Des exemples d'expressions sont présentés ci-après.

Exemple : égalité des chaînes

L'expression suivante sélectionne des instances avec le type d'instance spécifié.

```
attribute:ecs.instance-type == t2.small
```

Exemple : liste d'arguments

L'expression suivante sélectionne des instances dans la zone de disponibilité us-east-1a ou us-east-1b.

```
attribute:ecs.availability-zone in [us-east-1a, us-east-1b]
```

Exemple : expression composée

L'expression suivante sélectionne des instances G2 qui ne se trouvent pas dans la zone de disponibilité us-east-1d.

```
attribute:ecs.instance-type =~ g2.* and attribute:ecs.availability-zone != us-east-1d
```

Exemple : affinité des tâches

L'expression suivante sélectionne des instances qui hébergent des tâches dans le groupe `service:production`.

```
task:group == service:production
```

Exemple : anti-affinité des tâches

L'expression suivante sélectionne des instances qui n'hébergent pas de tâches dans le groupe de base de données.

```
not(task:group == database)
```

Exemple : nombre de tâches en cours d'exécution

L'expression suivante sélectionne des instances qui n'exécutent qu'une seule tâche.

```
runningTasksCount == 1
```

Exemple : version de l'agent de conteneur Amazon ECS

L'expression suivante sélectionne des instances qui exécutent une version de l'agent de conteneur antérieure à 1.14.5.

```
agentVersion < 1.14.5
```

Exemple : Date d'enregistrement d'instance

L'expression suivante sélectionne des instances qui ont été enregistrées avant le 13 février 2018.

```
registeredAt < 2018-02-13
```

Exemple : ID d' EC2 instance Amazon

L'expression suivante sélectionne des instances avec l' EC2 instance Amazon suivante IDs.

```
ec2InstanceId in ['i-abcd1234', 'i-wxyx7890']
```

Exemple de contraintes de placement de tâches Amazon ECS

Voici des exemples de contraintes de placement des tâches.

Cet exemple utilise la `memberOf` contrainte pour placer des tâches sur des instances t2. Il peut être spécifié à l'aide des actions suivantes : CreateServiceUpdateService, RegisterTaskDefinition, et RunTask.

```
"placementConstraints": [
    {
        "expression": "attribute:ecs.instance-type =~ t2.*",
        "type": "memberOf"
    }
]
```

L'exemple utilise la contrainte `memberOf` pour placer des tâches de réplica sur des instances avec des tâches dans le groupe de tâches `daemon-service` du service démon, en respectant les stratégies de placement des tâches qui sont également spécifiées. Cette contrainte garantit que les tâches de service du démon sont placées sur l' EC2 instance avant les tâches du service de réplication.

Remplacez `daemon-service` par le nom du service démon.

```
"placementConstraints": [
    {
        "expression": "task:group == service:daemon-service",
        "type": "memberOf"
    }
]
```

L'exemple utilise la contrainte `memberOf` pour placer des tâches sur des instances avec d'autres tâches dans le groupe de tâches `databases`, en respectant les stratégies de placement des tâches qui sont également spécifiées. Pour de plus amples informations sur les groupes de tâches, veuillez consulter Tâches Amazon ECS liées au groupe . Il peut être spécifié à l'aide des actions suivantes : CreateServiceUpdateService, RegisterTaskDefinition, et RunTask.

```
"placementConstraints": [
    {
        "expression": "task:group == databases",
        "type": "memberOf"
    }
```

```
]
```

La contrainte `distinctInstance` place chaque tâche du groupe sur une instance différente. Il peut être spécifié à l'aide des actions suivantes : CreateServiceUpdateService, et RunTask

Amazon ECS examine le statut souhaité des tâches pour le placement des tâches. Par exemple, si le statut souhaité de la tâche existante est`STOPPED`, (mais pas le dernier statut), une nouvelle tâche entrante peut être placée sur la même instance malgré la contrainte de `distinctInstance` placement. Par conséquent, vous pouvez voir deux tâches dont le dernier statut est `RUNNING` sur la même instance.

```
"placementConstraints": [
    {
        "type": "distinctInstance"
    }
]
```

Tâches autonomes Amazon ECS

Vous pouvez exécuter votre application en tant que tâche lorsqu'une application exécute un certain travail, puis s'arrête, par exemple un traitement par lots. Si vous souhaitez exécuter une tâche une seule fois, vous pouvez utiliser la console AWS CLI, APIs, ou SDKs.

Si vous devez exécuter votre application selon un calendrier basé sur un tarif, un cron ou un calendrier unique, vous pouvez créer un calendrier à l'aide du planificateur. EventBridge

Flux de travail des tâches

Lorsque vous lancez des tâches Amazon ECS (tâches autonomes ou par les services Amazon ECS), une tâche est créée et initialement déplacée vers l'`PROVISIONING`état. Lorsqu'une tâche est dans `PROVISIONING` cet état, ni la tâche ni les conteneurs n'existent car Amazon ECS a besoin de trouver la capacité de calcul pour placer la tâche.

Amazon ECS sélectionne la capacité de calcul appropriée pour votre tâche en fonction de votre type de lancement ou de la configuration de votre fournisseur de capacité. Vous pouvez utiliser des fournisseurs de capacité et des stratégies de fournisseurs de capacité avec les types de lancement Fargate et EC2 Amazon. Avec Fargate, vous n'avez pas à penser au provisionnement, à la configuration et au dimensionnement de la capacité de votre cluster. Fargate s'occupe de toute la gestion de l'infrastructure pour vos tâches. Pour le type de EC2 lancement, vous pouvez soit gérer la

capacité de votre cluster en enregistrant des EC2 instances Amazon dans votre cluster, soit utiliser le dimensionnement automatique du cluster pour simplifier la gestion de votre capacité de calcul. La mise à l'échelle automatique du cluster prend en charge le dimensionnement dynamique de la capacité de votre cluster, afin que vous puissiez vous concentrer sur l'exécution des tâches. Amazon ECS détermine où placer la tâche en fonction des exigences que vous spécifiez dans la définition de la tâche, telles que le processeur et la mémoire, ainsi que de vos contraintes et stratégies de placement. Pour plus d'informations, consultez, Comment Amazon ECS place les tâches sur les instances de conteneur.

Si vous utilisez un fournisseur de capacité sur lequel le dimensionnement géré est activé, les tâches qui ne peuvent pas être démarrées en raison d'un manque de capacité de calcul sont déplacées vers l'`PROVISIONING`état au lieu d'échouer immédiatement. Après avoir trouvé la capacité de placer votre tâche, Amazon ECS fournit les pièces jointes nécessaires (par exemple, Elastic Network Interfaces (ENIs) pour les tâches en `awsvpc` mode). Il utilise l'agent de conteneur Amazon ECS pour extraire les images de vos conteneurs, puis les démarrer. Une fois le provisionnement terminé et les conteneurs concernés lancés, Amazon ECS fait passer la tâche à `RUNNING` l'état. Pour plus d'informations sur les états des tâches, consultezCycle de vie des tâches Amazon ECS.

Exécution d'une application en tant que tâche Amazon ECS

Vous pouvez créer une tâche pour un processus ponctuel à l'aide du AWS Management Console.

Pour créer une tâche autonome ()AWS Management Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. La console Amazon ECS vous permet de créer une tâche autonome à partir de la page détaillée de votre cluster ou de la liste de révisions des définitions de tâches. Suivez les étapes ci-dessous pour créer votre tâche autonome en fonction de la page de ressources que vous avez choisie.

Pour démarrer un service à partir de	Étapes	
La page détaillée du cluster	a. Sur la page Clusters, sélectionnez le cluster dans lequel créer le service.	

Pour démarrer un service à partir de	Étapes	
	b. Dans l'onglet Tâches, choisissez Exécuter la tâche.	
La page de révision de la définition des tâches	a. Sur la page Définitions de tâches, choisissez la famille de définitions de tâches pour afficher les révisions de cette famille. b. Sélectionnez la révision que vous souhaitez utiliser. c. Dans le menu Déployer, choisissez Exécuter la tâche.	

3. Pour Cluster existant, choisissez le cluster.

   Choisissez Créer un cluster pour exécuter la tâche sur un nouveau cluster

4. Choisissez la manière dont vos tâches sont réparties dans l'infrastructure de votre cluster. Sous Configuration de calcul, choisissez votre option. Pour utiliser une stratégie de fournisseur de capacité, vous devez configurer vos fournisseurs de capacité au niveau du cluster.

   Si vous n'avez pas configuré votre cluster pour utiliser un fournisseur de capacité, utilisez plutôt un type de lancement.

Méthode de distribution	Étapes	
Stratégie de fournisseur de capacité	a. Dans Compute options (Options de calcul), sélectionnez Capacity provider strategy (Stratégie de fournisseur de capacité).	

Méthode de distribution	Étapes	
	b. Choisissez une stratégie : • Pour utiliser une stratégie de fournisseur de capacité, choisisse z Use cluster default (Utiliser la stratégie par défaut du cluster). • Si votre cluster ne possède pas de stratégie de fournisseur de capacité par défaut ou si vous souhaitez utiliser une stratégie personnalisée, sélection nez Use custom (Utiliser une stratégie personnal isée) ou Add capacity provider strategy (Ajouter une stratégie de fournisseur de capacité) et définisse z votre stratégie de fournisseur de capacité personnalisée en spécifiant une Base, un Capacity provider (Fournisseur de capacité) et un Weight (Poids). **Note** Pour utiliser un fournisseur de	

Méthode de distribution	Étapes
	capacité dans une stratégie, le fournisseur de capacité doit être associé au cluster.
Type de lancement	a. Dans Compute options (Options de calcul), sélectionnez Launch type (Type de lancement). b. Pour Launch type (Type de lancement), choisissez un type de lancement. c. (Facultatif) Lorsque le type de lancement Fargate est spécifié, pour Platform version (Version de plateforme), spécifiez la version de plateforme à utiliser. Si vous ne spécifiez aucune version de plateforme, la version `LATEST` est utilisée.

5. Sous Configuration du déploiement, procédez comme suit :

   a. Pour Définition de la tâche, entrez la définition de la tâche.

   > **Important**
   >
   > La console valide la sélection pour s'assurer que la famille et la révision de définition de tâche sélectionnées sont compatibles avec la configuration de calcul définie.

   b. Pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer.

   c. Pour Groupe de tâches, entrez le nom du groupe de tâches.

6. Si votre définition de tâche utilise le mode réseau `awsvpc`, développez Networking (Mise en réseau). Effectuez les étapes suivantes pour spécifier une configuration personnalisée.

   a. Pour VPC, sélectionnez le VPC à utiliser.

   b. Pour Subnets (Sous-réseaux), sélectionnez un ou plusieurs sous-réseaux du VPC que le planificateur de tâches prend en compte lors du placement de vos tâches.

   c. Pour Groupe de sécurité, vous pouvez choisir un groupe de sécurité existant ou en créer un nouveau. Pour utiliser un groupe de sécurité existant, sélectionnez-le et passez à l'étape suivante. Pour créer un nouveau groupe de sécurité, sélectionnez Create a new security group (Créer un nouveau groupe de sécurité). Vous devez spécifier un nom de groupe de sécurité et une description, puis ajouter une ou plusieurs règles entrantes pour le groupe de sécurité.

   d. Pour Public IP (Adresse IP publique), indiquez si vous voulez attribuer automatiquement une adresse IP publique à l'interface réseau Elastic (ENI) de la tâche.

      AWS Fargate les tâches peuvent se voir attribuer une adresse IP publique lorsqu'elles sont exécutées dans un sous-réseau public afin qu'elles disposent d'une route vers Internet. EC2 il n'est pas possible d'attribuer une adresse IP publique aux tâches à l'aide de ce champ. Pour plus d'informations, consultez les options de mise en réseau des tâches Amazon ECS pour le type de lancement Fargate et Allocation d'une interface réseau pour une tâche Amazon ECS. .

7. Si votre tâche utilise un volume de données compatible avec la configuration lors du déploiement, vous pouvez configurer le volume en développant Volume.

   Le nom et le type de volume sont configurés lors de la création d'une révision de définition de tâche et ne peuvent pas être modifiés lorsque vous exécutez une tâche autonome. Pour mettre à jour le nom et le type du volume, vous devez créer une nouvelle révision de définition de tâche et exécuter une tâche à l'aide de cette nouvelle révision.

Pour configurer ce type de volume	Faites ceci	
Amazon EBS	a. Pour le type de volume EBS, choisissez le type de volume EBS que vous souhaitez associer à votre tâche.	

Pour configurer ce type de volume	Faites ceci	
	b. Pour Taille (GiB), entrez une valeur valide pour la taille du volume en gibioctets (GiB). Vous pouvez spécifier une taille de volume minimale de 1 GiB et maximale de 16 384 GiB. Cette valeur est obligatoire sauf si vous fournissez un identifiant de capture d'écran. c. Pour les IOPS, entrez le nombre maximum d' input/output opérations (IOPS) que le volume doit fournir. Cette valeur est configurable uniquemen t pour les types de `gp3` volume `io1io2`, et. d. Pour Débit (Mib/s), entrez le débit que le volume doit fournir, en mégaoctets par seconde (ou Mib/s). MiBps Cette valeur est configura ble uniquement pour le type de `gp3` volume. e. Pour Snapshot ID, choisissez un instantané de volume Amazon EBS existant ou entrez l'ARN d'un instantané si vous souhaitez créer un volume à partir d'un instantané.	

Pour configurer ce type de volume	Faites ceci	
	Vous pouvez également créer un nouveau volume vide en ne choisissant ni en saisissant un identifiant de capture d'écran. f. Si vous spécifiez un ID de capture d'écran, vous pouvez spécifier un taux d'initialisation du volume (Mbits/s). Entrez une valeur comprise entre 100 et 300, en Mbits/s, qui déterminera la vitesse de chargement des données à partir de l'instantané spécifié à l'aide de l'ID de capture pour la création du volume. g. Pour la politique de résiliation, décochez la case si vous souhaitez que le volume configuré pour être attaché à la tâche soit préservé après la fin de la tâche. Par défaut, les volumes EBS attachés à des tâches sont supprimés lorsque la tâche est arrêtée. h. Pour Type de système de fichiers, choisissez le type de système de fichiers qui sera utilisé	

Pour configurer ce type de volume	Faites ceci	
	pour le stockage et la récupération des données sur le volume. Vous pouvez choisir le système d'exploitation par défaut ou un type de système de fichiers spécifique. La valeur par défaut pour Linux est`XFS`. Pour les volumes créés à partir d'un instantané, vous devez spécifier le même type de système de fichiers que celui utilisé par le volume lors de la création de l'instantané. Si le type de système de fichiers ne correspond pas, la tâche ne démarrera pas. i. Pour le rôle d'infrast ructure, choisissez un rôle IAM doté des autorisat ions nécessaires pour permettre à Amazon ECS de gérer les volumes Amazon EBS pour les tâches. Vous pouvez associer la politique `AmazonECSInfrastru ctureRole PolicyForVolumes` gérée au rôle, ou vous pouvez utiliser la politique	

Pour configurer ce type de volume	Faites ceci	
	comme guide pour créer et associer votre propre politique avec des autorisations répondant à vos besoins spécifiques. Pour plus d'informations sur les autorisations nécessaires, voir Rôle IAM dans l'infrastructure Amazon ECS. j. Pour le chiffrement, choisissez Par défaut si vous souhaitez utiliser le chiffrement Amazon EBS par défaut. Si le chiffrement est configuré par défaut sur votre compte, le volume sera chiffré avec la clé AWS Key Management Service (AWS KMS) spécifiée dans le paramètre. Si vous choisissez Par défaut et que le chiffrement par défaut d'Amazon EBS n'est pas activé, le volume ne sera pas chiffré. Si vous choisissez Personnalisé, vous pouvez spécifier celui AWS KMS key de votre	

Pour configurer ce type de volume	Faites ceci	
	choix pour le chiffrement du volume. Si vous choisissez Aucun, le volume ne sera pas chiffré, sauf si le chiffrement est configuré par défaut ou si vous créez un volume à partir d'un instantané chiffré. k. Si vous avez choisi Personnalisé pour le chiffrement, vous devez spécifier celui AWS KMS key que vous souhaitez utiliser. Pour la clé KMS, choisissez AWS KMS key ou entrez un ARN de clé. Si vous choisissez de chiffrer votre volume à l'aide d'une clé symétrique gérée par le client, assurez-vous que vous disposez des autorisations appropriées définies dans votre AWS KMS key politique. Pour plus d'informations, consultez la section Chiffrement des données pour les volumes Amazon EBS. l. (Facultatif) Sous Balises, vous pouvez ajouter des balises à votre volume	

Pour configurer ce type de volume	Faites ceci	
	Amazon EBS soit en propageant des balises à partir de la définition de la tâche, soit en fournissant vos propres balises. Si vous souhaitez propager des balises à partir de la définition de tâche, choisissez Définition de tâche pour propager des balises à partir de. Si vous choisissez Ne pas propager, ou si vous ne choisissez aucune valeur, les balises ne sont pas propagées. Si vous souhaitez fournir vos propres balises, choisissez Ajouter une balise, puis indiquez la clé et la valeur pour chaque balise que vous ajoutez. Pour plus d'informations sur le balisage des volumes Amazon EBS, consultez la section Marquage des volumes Amazon EBS.	

8. (Facultatif) Pour utiliser une stratégie de placement des tâches autre que la stratégie par défaut, développez Task Placement (Placement des tâches), puis choisissez parmi les options suivantes.

Pour de plus amples informations, veuillez consulter Comment Amazon ECS place les tâches sur les instances de conteneur.

- Répartition équilibrée AZ : répartissez les tâches entre les zones de disponibilité et entre les instances de conteneur de la zone de disponibilité.
- AZ Balanced BinPack — Répartissez les tâches entre les zones de disponibilité et entre les instances de conteneur disposant du moins de mémoire disponible.
- BinPack— Répartissez les tâches en fonction de la quantité minimale de processeur ou de mémoire disponible.
- Une tâche par hôte : placez au maximum une tâche du service sur chaque instance de conteneur.
- Personnalisé : définissez votre propre stratégie de placement des tâches.

Si vous avez choisi Custom (Personnaliser), définissez l'algorithme de placement des tâches et les règles à prendre en compte lors du placement des tâches.

- Sous Strategy (Stratégie), pour Type et Field (Champ), choisissez l'algorithme et l'entité à utiliser pour l'algorithme.

  Vous pouvez saisir jusqu'à 5 stratégies maximum.
- Sous Contrainte, pour Type et Expression, choisissez la règle et l'attribut pour la contrainte.

  Par exemple, pour définir la contrainte permettant de placer des tâches sur des instances T2, pour Expression, saisissez attribute:ecs.instance-type =~ t2.*.

  Vous pouvez saisir jusqu'à 10 contraintes maximum.

9. (Facultatif) Pour remplacer le rôle IAM de la tâche ou le rôle d'exécution de tâches défini dans votre définition de tâche, développezTask overrides (Remplacements de tâche), puis effectuez les étapes suivantes :

   a. Pour Rôle de tâche, choisissez un rôle IAM pour cette tâche. Pour de plus amples informations, veuillez consulter Rôle IAM de la tâche Amazon ECS.

      Seuls les rôles possédant une relation d'approbation `ecs-tasks.amazonaws.com` sont affichés. Pour savoir comment créer un rôle IAM pour vos tâches, consultez Création du rôle IAM de la tâche.

b. Pour Rôle d'exécution de tâche, choisissez un rôle d'exécution de tâche. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

10. (Facultatif) Pour remplacer les commandes du conteneur et les variables d'environnement, développez Container Overrides (Remplacements de conteneurs), puis développez le conteneur.

    - Pour envoyer une commande au conteneur autre que la commande de définition de tâche, dans Remplacement de commande, saisissez la commande Docker.
    - Pour ajouter une variable d'environnement, choisissez Add Environment Variable (Ajouter une variable d'environnement). Pour Key (Clé), saisissez le nom de votre variable d'environnement. Pour Value (Valeur), saisissez une valeur de chaîne pour la valeur d'environnement (sans guillemets doubles (`"  "`)).

      AWS entoure les chaînes de guillemets doubles (» «) et transmet la chaîne au conteneur au format suivant :

      ```
      MY_ENV_VAR="This variable contains a string."
      ```

11. (Facultatif) Pour vous aider à identifier votre tâche, développez Tags (balises), puis configurez vos balises.

    Pour qu'Amazon ECS étiquette automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises de définition des tâches, sélectionnez Turn on Amazon ECS managed tags (Activer les balises gérées par Amazon ECS), puis sélectionnez Task definitions (Définitions de tâches).

    Ajoutez ou supprimez une balise.

    - [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :
      - Pour Clé, saisissez le nom de la clé.
      - Pour Valeur, saisissez la valeur de clé.
    - [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.

12. Choisissez Créer.

Utilisation d'Amazon EventBridge Scheduler pour planifier des tâches Amazon ECS

EventBridge Le planificateur est un planificateur sans serveur qui vous permet de créer, d'exécuter et de gérer des tâches à partir d'un service géré centralisé. Il fournit une fonctionnalité de planification ponctuelle et récurrente indépendamment des bus et des règles de l'événement. EventBridge Le planificateur est hautement personnalisable et offre une évolutivité améliorée par rapport aux règles EventBridge planifiées, avec un ensemble plus large d'opérations et de services d'API cibles. AWS EventBridge Le planificateur fournit les plannings suivants que vous pouvez configurer pour vos tâches dans la console du EventBridge planificateur :

- Basé sur un taux
- Basées sur cron

  Vous pouvez configurer des planifications basées sur cron dans n'importe quel fuseau horaire.
- Planifications ponctuelles

  Vous pouvez configurer des planifications ponctuelles dans n'importe quel fuseau horaire.

Vous pouvez planifier votre Amazon ECS à l'aide d'Amazon EventBridge Scheduler.

Bien que vous puissiez créer une tâche planifiée dans la console Amazon ECS, la console EventBridge Scheduler fournit actuellement davantage de fonctionnalités.

Respectez les étapes suivantes avant de planifier une tâche :

1. Utilisez la console VPC pour obtenir le sous-réseau sur IDs lequel les tâches s'exécutent et le groupe de sécurité IDs pour les sous-réseaux. Pour plus d'informations, consultez les sections Sous-réseaux pour votre VPC et Contrôlez le trafic vers AWS vos ressources à l'aide de groupes de sécurité dans le guide de l'utilisateur Amazon VPC.
2. Configurez le EventBridge rôle d'exécution du planificateur. Pour plus d'informations, consultez la section Configurer le rôle d'exécution dans le guide de l'utilisateur d'Amazon EventBridge Scheduler.
3. Si vous souhaitez utiliser une stratégie de fournisseur de capacité pour exécuter la tâche, un fournisseur de capacité doit être associé au cluster.

Pour créer une planification à l'aide de la console

1. Ouvrez la console Amazon EventBridge Scheduler à https://console.aws.amazon.com/scheduler/la maison.
2. Sur la page Planifications, choisissez Créer une planification.
3. Sur la page Spécifier le détail de la planification, dans la section Nom et description de la planification, procédez comme suit :

   a. Pour Nom de la planification, saisissez un nom à attribuer à votre planification. Par exemple, **`MyTestSchedule`**.

   b. (Facultatif) Dans le champ Description, saisissez une description de la planification. Par exemple, **`TestSchedule`**.

   c. Pour Groupe de planification, choisissez un groupe de planification. Si vous n'avez pas de groupe, choisissez par défaut. Pour créer un groupe de planifications, choisissez Crée votre propre planification.

      Vous utilisez des groupes de planifications pour leur ajouter des balises.
4. Choisissez vos options de planification.

Occurrence	Faites ceci...
Planification ponctuelle Une planification ponctuelle n'invoque un objectif qu'une seule fois à la date et à l'heure que vous indiquez.	Pour Date et heure, procédez comme suit : • Entrez une date valide au format `YYYY/MM/DD` . • Entrez un horodatage au format `hh:mm` de 24 heures. • Dans le champ Fuseau horaire, choisissez le fuseau horaire.
Planification récurrente Une planification récurrente invoque un objectif à un taux	a. Pour Schedule type (Planifier le type), effectuez l'une des étapes suivantes :

Occurrence	Faites ceci...	
que vous spécifiez à l'aide d'une expression cron ou d'une expression rate.	• Pour utiliser une expression cron afin de définir la planification, choisissez Planification basée sur cron et entrez l'expression cron. • Pour utiliser une expression de rythme pour définir la planification, choisissez Planification basée sur le rythme. Pour plus d'informations sur les expressions cron et rate, consultez la section Types de planification sur EventBridge Scheduler dans le guide de l'utilisateur d'Amazon EventBridge Scheduler. b. Pour Fenêtre temporelle flexible, choisissez Désactivé pour désactiver cette option ou choisir l'une des fenêtres temporelles prédéfinies. Par exemple, si vous choisissez15 minutes et que vous définissez une planification récurrente pour invoquer son objectif une fois par	

Occurrence	Faites ceci...	
	heure, la planification s'exécute dans les 15 minutes suivant le début de chaque heure.	

5. (Facultatif) Si vous avez choisi Planification récurrente à l'étape précédente, dans la section Délai, procédez comme suit :
   a. Dans le champ Fuseau horaire, choisissez un fuseau horaire.
   b. Pour Date et heure de début, entrez une date valide au format `YYYY/MM/DD`, puis spécifiez un horodatage au format `hh:mm` de 24 heures.
   c. Pour Date et heure de fin, entrez une date valide au format `YYYY/MM/DD`, puis spécifiez un horodatage au format `hh:mm` de 24 heures.
6. Choisissez Suivant.
7. Sur la page Sélectionner une cible, procédez comme suit :
   a. Choisissez Tout APIs, puis entrez ECS dans la zone de recherche.
   b. Sélectionnez Amazon ECS.
   c. Dans la zone de recherche RunTask, entrez, puis choisissez RunTask.
   d. Pour Cluster ECS, choisissez le cluster.
   e. Pour Tâche ECS, choisissez la définition de tâche à utiliser pour la tâche.
   f. Choisissez la manière dont vos tâches sont réparties dans l'infrastructure de votre cluster. Développez les options de calcul, puis choisissez l'une des options suivantes

Option de calcul	Étapes	
Stratégie de fournisseur de capacité	a. Choisissez la stratégie du fournisseur de capacités. b. Choisissez une stratégie : • Pour utiliser la stratégie de fournisse ur de capacité par	

Option de calcul	Étapes	
	défaut, choisissez Utiliser la stratégie par défaut du cluster. • Pour utiliser une stratégie personnal isée, choisissez Utiliser une stratégie personnalisée. Entrez ensuite la base, le fournisseur de capacité et le poids. En effet EC2, le fournisseur de capacité est le groupe Auto Scaling.	
Type de lancement	a. Choisissez le type de lancement. b. Pour Launch type (Type de lancement), choisisse z un type de lancement. c. Lorsque le type de lancement Fargate est spécifié, pour la version de plate-forme, spécifiez la version de plate-forme à utiliser.	

g. Pour les sous-réseaux, entrez le sous-réseau dans IDs lequel exécuter la tâche.

h. Pour les groupes de sécurité, entrez le groupe IDs de sécurité du sous-réseau.

i. (Facultatif) Pour utiliser une stratégie de placement des tâches autre que la stratégie par défaut, développez Contrainte de placement, puis saisissez les contraintes.

Pour de plus amples informations, veuillez consulter Comment Amazon ECS place les tâches sur les instances de conteneur.

j. (Facultatif) Pour vous aider à identifier vos tâches, sous Balises, configurez vos balises.

Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec les balises de définition des tâches, sélectionnez Activer les balises gérées par Amazon ECS.

8. Choisissez Suivant.
9. Sur la page Settings (Paramètres), procédez comme suit :

a. Pour activer la planification, sous État de la planification, activez Activer la planification.

b. Pour configurer une stratégie de nouvelles tentatives pour votre planification, sous Politique de nouvelle tentative et file d'attente de lettres mortes (DLQ), procédez comme suit :

- Activez Réessayer.
- Pour Durée maximale de rétention de l'événement, entrez le nombre maximum d'heures et de minutes pendant lequel le EventBridge planificateur doit conserver un événement non traité.
- La durée maximale est 24 heures.
- Pour Nombre maximum de tentatives, entrez le nombre maximum de fois que le EventBridge planificateur réessaie le calendrier si la cible renvoie une erreur.

  La valeur maximale est 185 nouvelles tentatives.

Avec les politiques de nouvelle tentative, si un calendrier ne parvient pas à invoquer sa cible, le EventBridge planificateur le réexécute. Si elle est configurée, vous devez définir la durée de rétention maximale et les nouvelles tentatives pour la planification.

c. Choisissez l'endroit où EventBridge Scheduler stocke les événements non livrés.

Option File d'attente de lettres mortes (DLQ)	Faites ceci...
Ne stockez pas	Sélectionnez Aucun.

Option File d'attente de lettres mortes (DLQ)	Faites ceci...
Enregistrez l'événement dans le même Compte AWS endroit où vous créez le calendrier	a. Choisissez Sélectionnez une file d'attente Amazon SQS dans my Compte AWS as a DLQ. b. Choisissez l'Amazon Resource Name (ARN) de la file d'attente Amazon SQS.
Stockez l'événement dans un endroit Compte AWS différent de celui dans lequel vous créez le calendrier	a. Choisissez Spécifier une file d'attente Amazon SQS dans un autre en Comptes AWS tant que DLQ. b. Entrez l'Amazon Resource Name (ARN) de la file d'attente Amazon SQS.

d. Pour utiliser une clé gérée par le client afin de chiffrer votre entrée cible, sous Chiffrement, choisissez Personnaliser les paramètres de chiffrement (avancé).

   Si vous choisissez cette option, entrez un ARN de clé KMS existant ou choisissez Créez un AWS KMS keypour accéder à la console AWS KMS . Pour plus d'informations sur la manière dont EventBridge Scheduler chiffre vos données au repos, consultez la section Chiffrement au repos dans le guide de l'utilisateur d'Amazon EventBridge Scheduler.

e. Pour Autorisations, choisissez Utiliser le rôle existant, puis sélectionnez le rôle.

   Pour que le EventBridge planificateur crée un nouveau rôle d'exécution pour vous, choisissez Créer un nouveau rôle pour ce calendrier. Ensuite, saisissez un nom pour Nom du rôle. Si vous choisissez cette option, le EventBridge planificateur associe au rôle les autorisations requises pour votre cible modélisée.

10. Choisissez Suivant.

11. Sur la page Examiner et créer une planification, examinez les détails de votre planification. Dans chaque section, choisissez Modifier pour revenir à cette étape et modifier ses détails.
12. Choisissez Créer une planification.

    Vous pouvez consulter la liste de vos planifications nouvelles et existantes sur la page Planifications. Sous la colonne État, vérifiez que votre nouvelle planification est activée.

Étapes suivantes

Vous pouvez utiliser la console du EventBridge planificateur ou le AWS CLI pour gérer le calendrier. Pour plus d'informations, consultez la section Gérer un calendrier dans le guide de l'utilisateur d'Amazon EventBridge Scheduler.

Arrêt d'une tâche Amazon ECS

Si vous n'avez plus besoin de maintenir une tâche autonome en cours d'exécution, vous pouvez l'arrêter. La console Amazon ECS permet d'arrêter facilement une ou plusieurs tâches.

Vous ne pouvez pas redémarrer une tâche arrêtée de manière autonome.

Si vous souhaitez arrêter un service, consultezSupprimer un service Amazon ECS à l'aide de la console.

Pour arrêter une tâche autonome ()AWS Management Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster pour accéder à la page de détails du cluster.
4. Sur la page détaillée du cluster, choisissez l'onglet Tâches.
5. Vous pouvez filtrer les tâches par type de lancement à l'aide de la liste des types de lancement du filtre.

Tâches à arrêter	Étapes	
Un ou plusieurs	a. Sélectionnez les tâches, puis choisissez Arrêter, Arrêter sélectionné.	

Tâches à arrêter	Étapes
	b. Sur la page de confirmation de l'arrêt de la tâche, choisissez Arrêter
Tous	**Important** Si vous choisissez d'arrêter toutes les tâches à l'aide de la console, Amazon ECS arrête toutes les tâches autonomes et celles qui font partie d'un service. Par conséquent, nous vous recommandons de faire preuve de prudence lorsque vous utilisez cette option. a. Choisissez Arrêter, Arrêter tout. b. Sur la page de confirmation Arrêter les tâches, entrez Arrêter toutes les tâches, puis sélectionnez Arrêter.

Services Amazon ECS

Vous pouvez utiliser un service Amazon ECS service pour exécuter et gérer simultanément un nombre spécifié d'instances d'une définition de tâche dans un cluster Amazon ECS. Si l'une de

vos tâches échoue ou s'arrête, le planificateur de service d'Amazon ECS service lance une autre instance de votre définition de tâche pour la remplacer. Cela permet de maintenir le nombre de tâches souhaité dans le service.

Le cas échéant, vous pouvez également exécuter votre service derrière un équilibreur de charge. Cet équilibreur de charge répartit le trafic sur les tâches associées au service.

Nous vous recommandons d'utiliser le planificateur de service pour les services et applications sans état de longue durée. Le planificateur de service garantit que la stratégie de planification que vous spécifiez est suivie et replanifie les tâches en cas d'échec de l'une d'elles. Par exemple, si l'infrastructure sous-jacente échoue, le planificateur de services replanifie une tâche. Vous pouvez utiliser des stratégies et des contraintes de placement des tâches afin de personnaliser la façon dont le planificateur place et résilie les tâches. Si une tâche d'un service s'arrête, le planificateur lance une nouvelle tâche pour la remplacer. Ce processus se poursuit jusqu'à ce que votre service atteigne le nombre de tâches souhaité en fonction de la stratégie de planification que le service utilise. La stratégie de planification du service est également appelée type de service.

Le planificateur de services remplace également les tâches jugées défectueuses après l'échec d'une surveillance de l'état du conteneur ou du groupe cible de l'équilibreur de charge. Ce remplacement dépend des paramètres de définition du service `maximumPercent` et `desiredCount`. Si une tâche est marquée comme défectueuse, le planificateur de services lance d'abord une tâche de remplacement. Ensuite, ce qui suit se produit.

- Si l'état de santé de la tâche de remplacement est égal à`HEALTHY`, le planificateur de services arrête la tâche défectueuse
- Si l'état de santé de la tâche de remplacement est `UNHEALTHY`, le planificateur arrête soit la tâche de remplacement défectueuse, soit la tâche défectueuse existante pour que le nombre total de tâches soit égal à `desiredCount`.

Si le paramètre `maximumPercent` empêche le planificateur de démarrer d'abord une tâche de remplacement, le planificateur arrête une par une les tâches défectueuses au hasard pour libérer de la capacité, puis lance une tâche de remplacement. Le processus de démarrage et d'arrêt se poursuit jusqu'à ce que toutes les tâches défectueuses soient remplacées par des tâches saines. Une fois que toutes les tâches défectueuses ont été remplacées et que seules les tâches saines sont en cours d'exécution, si le nombre total de tâches dépasse le `desiredCount`, les tâches saines sont arrêtées au hasard jusqu'à ce que le nombre total de tâches soit égal à `desiredCount`. Pour plus d'informations sur `maximumPercent` et `desiredCount`, veuillez consulter Paramètres de définition de service (langue française non garantie).

Le planificateur de service inclut une logique qui limite la fréquence de redémarrage des tâches en cas d'échec de démarrage répété. Si une tâche est arrêtée sans avoir saisi d'état `RUNNING`, le planificateur de service commence à réduire les tentatives de lancement et envoie un message d'événement de service. Ce comportement évite d'utiliser inutilement les ressources pour des tâches qui ont échoué avant que vous puissiez résoudre le problème. Une fois le service mis à jour, le planificateur de service se comporte de nouveau normalement. Pour plus d'informations, consultez Logique de régulation du service Amazon ECS et Affichage des messages relatifs aux événements du service Amazon ECS.

Option de calcul d'infrastructure

Deux options de calcul permettent de distribuer vos tâches.

- Une capacity provider strategy (stratégie de fournisseurs de capacités) permet à Amazon ECS de distribuer vos tâches avec un ou plusieurs fournisseurs de capacité.

  Pour la stratégie des fournisseurs de capacités, la console sélectionne une option de calcul par défaut. Voici l'ordre utilisé par la console pour sélectionner une valeur par défaut :

  - Si votre cluster a une stratégie de fournisseur de capacité définie par défaut, elle est sélectionnée.
  - Si aucune stratégie de fournisseur de capacité par défaut n'est définie pour votre cluster mais que les fournisseurs de capacité Fargate y ont été ajoutés, une stratégie de fournisseur de capacité personnalisée utilisant `FARGATE` le fournisseur de capacité est sélectionnée.
  - Si aucune stratégie de fournisseur de capacité par défaut n'est définie pour votre cluster, mais qu'un ou plusieurs fournisseurs de capacité de groupe Auto Scaling y ont été ajoutés, l'option Use custom (Advanced) est sélectionnée et vous devez définir manuellement la stratégie.
  - Si votre cluster n'a pas de stratégie de fournisseur de capacité par défaut définie et qu'aucun fournisseur de capacité n'est ajouté au cluster, le type de lancement Fargate est sélectionné.
- Un type de lancement permet à Amazon ECS de lancer nos tâches directement sur Fargate ou sur EC2 les instances enregistrées dans vos clusters.

  Par défaut, le service démarre dans les sous-réseaux de votre VPC de cluster.

Service Auto Scaling

Le dimensionnement automatique des services permet d'augmenter ou de diminuer automatiquement le nombre de tâches souhaité dans votre service Amazon ECS. Amazon ECS utilise le service Application Auto Scaling pour fournir cette fonctionnalité.

Pour de plus amples informations, veuillez consulter Faites évoluer automatiquement votre service Amazon ECS.

Équilibrage de charge des services

Les services Amazon ECS hébergés sur ce site AWS Fargate prennent en charge les équilibreurs de charge d'application, les équilibreurs de charge réseau et les équilibreurs de charge de passerelle. Consultez le tableau suivant pour savoir quel type d'équilibreur de charge utiliser.

Type de Load Balancer	À utiliser dans ces cas	
Application Load Balancer	Trafic routier HTTP/HTTPS (ou couche 7). Les équilibreurs de charge Application Load Balancer offrent plusieurs fonctions qui les rendent intéressants à utiliser avec les services Amazon ECS service : • Chaque service peut desservir le trafic provenant de plusieurs équilibreurs de charge et exposer plusieurs ports à charge équilibrée en spécifiant plusieurs groupes cibles. • Ils sont pris en charge par des tâches hébergées à la fois sur EC2 Fargate et sur des instances.	

Type de Load Balancer	À utiliser dans ces cas	
	• Les Application Load Balancers permettent aux conteneurs d'utiliser le mappage de port hôte dynamique (de sorte que plusieurs tâches du même service soient autorisées par instance de conteneur). • Les Application Load Balancers prennent en charge le routage basé sur le chemin d'accès et les règles de priorité (pour que plusieurs services puissent utiliser le même port d'écoute sur un même Application Load Balancer).	
Network Load Balancer	Acheminez le trafic TCP ou UDP (ou couche 4).	
Passerelle équilibreur de charge	Acheminez le trafic TCP ou UDP (ou couche 4). Utilisez des appareils virtuels, tels que des pare-feux, des systèmes de détection et de prévention des intrusions et des systèmes d'inspection approfondie des paquets.	

Pour de plus amples informations, veuillez consulter Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS.

Services d'interconnexion

Si vous avez besoin d'une application pour vous connecter à d'autres applications qui s'exécutent en tant que services Amazon ECS, Amazon ECS propose les méthodes suivantes pour le faire sans équilibreur de charge :

- Service Connect - Permet les service-to-service communications avec découverte automatique à l'aide de noms abrégés et de ports standard.
- Découverte de services : la découverte de services utilise Route 53 pour créer un espace de noms pour votre service, ce qui permet de le découvrir via le DNS.
- Amazon VPC Lattice - VPC Lattice est un service de mise en réseau d'applications entièrement géré permettant de connecter, de sécuriser et de surveiller vos services sur plusieurs comptes et. VPCs Il y a un coût qui y est associé.

Pour de plus amples informations, veuillez consulter Interconnectez les services Amazon ECS.

Contrôles et stratégies de déploiement des services Amazon ECS

Avant de déployer votre service, déterminez les options de déploiement et les fonctionnalités utilisées par le service.

Stratégie de planification

Deux stratégies de planificateur de service sont disponibles :

- `REPLICA` – La stratégie de planification des réplicas place et gère le nombre de tâches souhaité dans votre cluster. Par défaut, le planificateur de service répartit les tâches entre les zones de disponibilité. Vous pouvez utiliser des stratégies et des contraintes de placement des tâches afin de personnaliser la façon dont les tâches sont placées. Pour de plus amples informations, veuillez consulter Stratégie de planification des répliques.
- `DAEMON` – La stratégie de planification du démon déploie exactement une tâche sur chaque instance de conteneur active qui répond à toutes les contraintes de placement des tâches spécifiées dans votre cluster. Lors de l'utilisation de cette stratégie, il n'est pas nécessaire de spécifier un nombre de tâches souhaité, une stratégie de placement des tâches ou d'utiliser les politiques Service Auto Scaling. Pour de plus amples informations, veuillez consulter Stratégie de planification des démons.

> **Note**
>
> Les tâches Fargate ne prennent pas en charge la stratégie de planification `DAEMON`.

Stratégie de planification des répliques

La stratégie de planification de réplica place et gère le nombre de tâches souhaité dans votre cluster.

Pour un service qui exécute des tâches sur Fargate, lorsque le planificateur de service lance de nouvelles tâches ou cesse l'exécution des tâches, le planificateur de service tente de maintenir un équilibre entre les zones de disponibilité de votre service. Vous n'avez pas besoin de préciser les stratégies ou les contraintes de placement des tâches.

Lorsque vous créez un service qui exécute des tâches sur des EC2 instances, vous pouvez éventuellement définir des stratégies et des contraintes de placement des tâches afin de personnaliser les décisions relatives au placement des tâches. Si aucune stratégie ou contrainte de placement de tâche n'est spécifiée, le planificateur de service répartit les tâches entre les zones de disponibilité par défaut. Le planificateur de service utilise la logique suivante :

- Détermine les instances de conteneur de votre cluster susceptibles de prendre en charge la définition de tâche de votre service (par exemple, avec le processeur, la mémoire, les ports et les attributs d'instances de conteneur exigés).
- Détermine les instances de conteneur qui respectent les contraintes de placement éventuellement définies pour le service.
- Lorsqu'un service de réplication dépend d'un service démon (par exemple, une tâche daemon log router qui doit être exécutée avant que les tâches puissent utiliser la journalisation), créez une contrainte de placement des tâches qui garantit que les tâches du service daemon sont placées sur l' EC2 instance avant les tâches du service de réplique. Pour de plus amples informations, veuillez consulter Exemple de contraintes de placement de tâches Amazon ECS.
- Lorsqu'une stratégie de placement est définie, utilisez-la pour sélectionner une instance parmi les candidats restants.
- Lorsqu'aucune stratégie de placement n'est définie, utilisez la logique suivante pour équilibrer les tâches entre les zones de disponibilité (Availability Zones) de votre cluster :
  - Trie les instances de conteneur valides. Accorde la priorité aux instances ayant le plus petit nombre de tâches en cours d'exécution pour ce service dans leur zone de disponibilité

respective. Par exemple, si la zone A comporte une tâche de service en cours d'exécution tandis que les zones B et C n'en ont pas, les instances de conteneur valides des zones B ou C sont considérées comme optimales pour le placement.

- Place la nouvelle tâche de service sur une instance de conteneur valide dans une zone de disponibilité optimale basée sur les étapes précédentes. Privilégie les instances de conteneur ayant le plus petit nombre de tâches en cours d'exécution pour ce service.

Nous vous recommandons d'utiliser la fonctionnalité de rééquilibrage des services lorsque vous utilisez la `REPLICA` stratégie, car elle permet de garantir la haute disponibilité de votre service.

Stratégie de planification des démons

La stratégie de planification du démon déploie exactement une tâche sur chaque instance de conteneur active qui répond à toutes les contraintes de placement des tâches spécifiées dans votre cluster. Le planificateur de services évalue les contraintes de placement des tâches pour les tâches en cours d'exécution et arrête les tâches qui ne répondent pas aux contraintes de placement. Lorsque vous utilisez cette stratégie, vous n'avez pas besoin de spécifier le nombre de tâches souhaité, de stratégie de placement des tâches ou d'utiliser les politiques de Service Auto Scaling.

Amazon ECS réserve des ressources de calcul d'instance de conteneur, y compris le CPU, la mémoire et l'interface réseau pour les tâches de démon. Lorsque vous lancez un service de démon sur un cluster avec d'autres services de réplica, Amazon ECS accorde la priorité à la tâche de démon. Cela signifie que la tâche daemon est la première tâche à être lancée sur les instances et la dernière à s'arrêter une fois que toutes les tâches de réplication ont été arrêtées. Cette stratégie garantit que les ressources ne sont pas utilisées par les tâches de réplica en attente et sont disponibles pour les tâches de démon.

Le planificateur de service du démon ne place aucune tâche sur les instances qui ont un statut `DRAINING`. Si une instance de conteneur passe au statut `DRAINING`, les tâches du démon sont arrêtées. Le planificateur de service surveille également l'ajout de nouvelles instances de conteneur à votre cluster et leur ajoute les tâches du démon.

Lorsque vous spécifiez une configuration de déploiement, la valeur du `maximumPercent` paramètre doit être `100` (spécifiée sous forme de pourcentage), qui est la valeur par défaut utilisée si elle n'est pas définie. La valeur par défaut du `minimumHealthyPercent` paramètre est `0` (spécifiée sous forme de pourcentage).

Vous devez redémarrer le service lorsque vous modifiez les contraintes de placement pour le service de démon. Amazon ECS met à jour dynamiquement les ressources qui sont réservées sur les

instances éligibles pour la tâche démon. Pour les instances existantes, le planificateur tente de placer la tâche sur l'instance.

Un nouveau déploiement démarre lorsqu'il existe une modification de la taille de la tâche ou de la réservation de ressource de conteneur dans la définition de tâche. Un nouveau déploiement démarre également lors de la mise à jour d'un service ou lors de la définition d'une révision différente de la définition de tâche. Amazon ECS récupère les réservations de CPU et de mémoire mises à jour pour le démon, puis bloque cette capacité pour la tâche démon.

Si les ressources sont insuffisantes pour l'un ou l'autre des cas ci-dessus, l'une des situations suivantes se produit :

- Le placement de la tâche échoue.
- Un CloudWatch événement est généré.
- Amazon ECS continue d'essayer de planifier la tâche sur l'instance en attendant que les ressources soient disponibles.
- Amazon ECS libère toutes les instances réservées qui ne répondent plus aux critères de contrainte de placement et arrête les tâches démon correspondantes.

La stratégie de planification de démon peut être utilisée dans les cas suivants :

- Exécution de conteneurs d'applications
- Exécution de conteneurs de support pour les tâches de journalisation, de surveillance et de suivi

Les tâches qui utilisent le type de lancement Fargate ou les types de contrôleur de déploiement `CODE_DEPLOY` ou `EXTERNAL` ne prennent pas en charge la stratégie de planification de démon.

Lorsque le planificateur de service arrête d'exécuter les tâches, il tente de conserver un équilibre entre les zones de disponibilité de votre cluster. Le planificateur utilise la logique suivante :

- Si une stratégie de placement est définie, utilisez cette stratégie pour sélectionner les tâches à résilier. Par exemple, si une stratégie de répartition par zone de disponibilité est définie pour un service, une tâche est sélectionnée, laissant ainsi aux tâches restantes la meilleure répartition.
- Si aucune stratégie de placement n'est définie, maintenez l'équilibre entre les zones de disponibilité de votre cluster selon la logique suivante :
  - Triez les instances de conteneur valides. Accordez la priorité aux instances ayant le plus grand nombre de tâches en cours d'exécution pour ce service dans leur zone de disponibilité

respective. Par exemple, si la zone A comporte une tâche de service en cours d'exécution tandis que les zones B et C en ont chacune deux tâches de service en cours d'exécution, les instances de conteneur des zones B ou C sont considérées comme optimales pour la mise hors service.

- Arrêtez la tâche sur une instance de conteneur dans une zone de disponibilité optimale basée sur les étapes précédentes. Privilégier les instances de conteneur ayant le plus grand nombre de tâches en cours d'exécution pour ce service.

Contrôleurs de déploiement

Le contrôleur de déploiement est le mécanisme qui détermine la manière dont les tâches sont déployées pour votre service. Les options valides sont les suivantes :

- ECS

  Lorsque vous créez un service qui utilise le contrôleur de `ECS` déploiement, vous pouvez choisir entre les stratégies de déploiement suivantes :

  - `ROLLING`: Lorsque vous créez un service qui utilise la stratégie de déploiement rolling update (`ROLLING`), le planificateur de services Amazon ECS remplace les tâches en cours d'exécution par de nouvelles tâches. Le nombre de tâches ajoutées au service ou supprimées du service par Amazon ECS lors d'une mise à jour propagée est contrôlé par la configuration de déploiement du service.

    Les déploiements de mises à jour continues sont les mieux adaptés aux scénarios suivants :

    - Mises à jour progressives du service : vous devez mettre à jour votre service progressivement sans mettre l'ensemble du service hors ligne en une seule fois.
    - Besoins en ressources limités : vous souhaitez éviter les coûts de ressources supplémentaires liés à l'exécution simultanée de deux environnements complets (comme l'exigent les déploiements bleu/vert).
    - Temps de déploiement acceptable : votre application peut tolérer un processus de déploiement plus long, car les mises à jour continues remplacent les tâches une par une.
    - Pas besoin d'annulation instantanée : votre service peut tolérer un processus de restauration qui prend quelques minutes au lieu de quelques secondes.
    - Processus de déploiement simple : vous préférez une approche de déploiement directe, sans la complexité liée à la gestion de plusieurs environnements, groupes cibles et auditeurs.

- Aucun équilibreur de charge requis : votre service n'utilise ni n'exige d'équilibreur de charge, d'Application Load Balancer, de Network Load Balancer ou de Service Connect (qui sont nécessaires pour les déploiements). blue/green
- Applications dynamiques : votre application conserve un état qui rend difficile l'exécution de deux environnements parallèles.
- Sensibilité aux coûts : vous souhaitez minimiser les coûts de déploiement en évitant d'exécuter des environnements dupliqués pendant le déploiement.

Les mises à jour continues constituent la stratégie de déploiement par défaut des services et fournissent un équilibre entre la sécurité du déploiement et l'efficacité des ressources pour de nombreux scénarios d'applications courants.

- `BLUE_GREEN`: Une stratégie de déploiement bleu/vert (`BLUE_GREEN`) est une méthodologie de publication qui réduit les temps d'arrêt et les risques en exécutant deux environnements de production identiques appelés bleu et vert. Avec les blue/green déploiements Amazon ECS, vous pouvez valider les nouvelles révisions des services avant de diriger le trafic de production vers celles-ci. Cette approche fournit un moyen plus sûr de déployer les modifications avec la possibilité de revenir rapidement en arrière si nécessaire.

  Les blue/green déploiements Amazon ECS sont parfaitement adaptés aux scénarios suivants :

  - Validation des services : lorsque vous devez valider de nouvelles révisions de service avant de diriger le trafic de production vers celles-ci
  - Aucun temps d'arrêt : lorsque votre service nécessite des déploiements sans interruption
  - Annulation instantanée : lorsque vous avez besoin de pouvoir revenir rapidement en arrière si des problèmes sont détectés
  - Exigence d'un équilibreur de charge : lorsque votre service utilise Application Load Balancer, Network Load Balancer ou Service Connect
- Externe

  Utilisez un contrôleur de déploiement tiers.
- Déploiement bleu/vert (alimenté par) AWS CodeDeploy

  CodeDeploy installe une version mise à jour de l'application en tant que nouvel ensemble de tâches de remplacement et redirige le trafic de production de l'ensemble de tâches d'application d'origine vers le jeu de tâches de remplacement. L'ensemble de tâches d'origine est arrêté une fois le déploiement terminé. Utilisez ce contrôleur de déploiement pour vérifier le nouveau déploiement d'un service avant de lui envoyer du trafic de production.

Détection des échecs de déploiement d'Amazon ECS

Amazon ECS propose deux méthodes pour détecter les échecs de déploiement :

- Disjoncteur de déploiement
- CloudWatch Alarmes

Les deux méthodes peuvent être configurées pour rétablir automatiquement le dernier bon état connu des déploiements ayant échoué.

Éléments à prendre en compte :

- Les deux méthodes ne prennent en charge que le déploiement de mises à jour continues et les types de blue/green déploiement.
- Lorsque les deux méthodes sont utilisées, l'une ou l'autre peut provoquer un échec du déploiement.
- La méthode de restauration nécessite un déploiement antérieur à l'état COMPLETED.
- EventBridge des événements sont générés pour les modifications de l'état du déploiement.

Comment le disjoncteur de déploiement Amazon ECS détecte les défaillances

Le disjoncteur du circuit de déploiement est un mécanisme de mise à jour propagée qui détermine si les tâches atteignent un état stable. Le disjoncteur du circuit de déploiement dispose d'une option qui ramène automatiquement un déploiement ayant échoué à l'état `COMPLETED`.

Lorsqu'un déploiement de service change d'état, Amazon ECS envoie un événement de changement d'état de déploiement de service à EventBridge. Cela permet de surveiller l'état de vos déploiements de services par programmation. Pour de plus amples informations, veuillez consulter Événements de changement d'état du déploiement du service Amazon ECS. Nous vous recommandons de créer et de surveiller une EventBridge règle avec un `eventName` de `SERVICE_DEPLOYMENT_FAILED` afin de pouvoir effectuer une action manuelle pour démarrer votre déploiement. Pour plus d'informations, consultez Getting Started with EventBridge dans le guide de EventBridge l'utilisateur Amazon.

Lorsque le disjoncteur du circuit de déploiement détermine qu'un déploiement a échoué, il recherche le déploiement le plus récent à l'état `COMPLETED`. Il s'agit du déploiement qu'il utilise comme déploiement de restauration. Lorsque la restauration commence, le déploiement passe de `COMPLETED` à `IN_PROGRESS`. Cela signifie que le déploiement n'est pas éligible à une autre

annulation tant qu'il n'a pas atteint un `COMPLETED` état. Lorsque le disjoncteur du circuit de déploiement ne trouve aucun déploiement à l'état `COMPLETED`, il ne lance pas de nouvelles tâches et le déploiement est bloqué.

Lorsque vous créez un service, le planificateur enregistre les tâches qui n'ont pas pu être lancées en deux étapes.

- Étape 1 - Le planificateur surveille les tâches pour voir si elles passent à l'état RUNNING.
  - Succès : le déploiement a une chance de passer à l'état COMPLETED car plusieurs tâches sont passées à l'état RUNNING. Les critères de défaillance sont ignorés et le disjoncteur passe à l'étape 2.
  - Échec : certaines tâches consécutives ne sont pas passées à l'état RUNNING et le déploiement peut passer à l'état FAILED.
- Étape 2 - Le déploiement entre dans cette phase lorsqu'au moins une tâche est en cours d'exécution. Le disjoncteur vérifie les bilans de santé des tâches du déploiement en cours d'évaluation. Les contrôles de santé validés sont Elastic Load Balancing, les contrôles de santé des AWS Cloud Map services et les contrôles de santé des conteneurs.
  - Réussite : au moins une tâche est en cours d'exécution et les tests de santé ont été validés.
  - Défaillance : les tâches remplacées en raison d'échecs liés aux tests de santé ont atteint le seuil d'échec.

Tenez compte des points suivants lorsque vous utilisez la méthode du disjoncteur de déploiement sur un service. EventBridge génère la règle.

- La réponse `DescribeServices` donne un aperçu de l'état d'un déploiement, le `rolloutState` et `rolloutStateReason`. Lorsqu'un nouveau déploiement est démarré, il commence avec l'état `IN_PROGRESS`. Lorsque le service atteint un état stable, l'état du déploiement passe à `COMPLETED`. Si le service n'arrive pas à atteindre un état stable et que le disjoncteur de circuit est activé, le déploiement passe à l'état `FAILED`. Un déploiement dans un état `FAILED` ne lance aucune nouvelle tâche.
- Outre les événements de changement d'état de déploiement de service qu'Amazon ECS envoie pour les déploiements qui ont démarré et se sont terminés, Amazon ECS envoie également un événement lorsqu'un déploiement avec le disjoncteur de circuit activé échoue. Ces événements fournissent des informations sur la raison pour laquelle un déploiement a échoué ou si un déploiement a été démarré en raison d'une restauration. Pour de plus amples informations, veuillez consulter Événements de changement d'état du déploiement du service Amazon ECS.

- Si un nouveau déploiement est démarré parce qu'un déploiement précédent a échoué et que la restauration a eu lieu, le champ `reason` de l'événement de changement d'état de déploiement de service indique que le déploiement a été démarré en raison d'une restauration.
- Le disjoncteur de circuit de déploiement est pris en charge uniquement pour les services Amazon ECS qui utilisent le contrôleur de déploiement de la mise à jour continue (`ECS`).
- Vous devez utiliser la console Amazon ECS, ou AWS CLI lorsque vous utilisez le disjoncteur de déploiement avec l' CloudWatch option. Pour plus d'informations, consultez the section called "Créer un service à l'aide de paramètres définis" et create-service dans la Référence AWS Command Line Interface .

L'`create-service` AWS CLI exemple suivant montre comment créer un service Linux lorsque le disjoncteur de déploiement est utilisé avec l'option de restauration.

```
aws ecs create-service \
     --service-name MyService \
     --deployment-controller type=ECS \
     --desired-count 3 \
     --deployment-configuration "deploymentCircuitBreaker={enable=true,rollback=true}"
 \
     --task-definition sample-fargate:1 \
     --launch-type FARGATE \
     --platform-family LINUX \
     --platform-version 1.4.0 \
     --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
```

Exemple :

Le déploiement 1 est à l'état `COMPLETED`.

Le déploiement 2 ne pouvant pas démarrer, le disjoncteur du circuit revient au déploiement 1. Le déploiement 1 passe à l'état `IN_PROGRESS`.

Le déploiement 3 démarre et aucun déploiement n'est à l'état `COMPLETED`. Le déploiement 3 ne peut donc ni être restauré, ni lancer de tâches.

Seuil d'échec

Le disjoncteur de déploiement calcule la valeur de seuil, puis utilise cette valeur pour déterminer quand passer le déploiement à l'état `FAILED`.

Le disjoncteur de déploiement a un seuil minimum de 3 et un seuil maximum de 200. Il utilise les valeurs de la formule suivante pour déterminer l'échec du déploiement.

```
Minimum threshold <= 0.5 * desired task count => maximum threshold
```

Lorsque le résultat du calcul est supérieur au minimum de 3, mais inférieur au maximum de 200, le seuil de défaillance est défini sur le seuil calculé (arrondi au chiffre supérieur).

> **Note**
>
> Vous ne pouvez modifier aucune des valeurs de seuil.

La surveillance de l'état du déploiement comporte deux étapes.

1. Le disjoncteur de déploiement surveille les tâches qui font partie du déploiement et vérifie les tâches qui se trouvent dans l'état `RUNNING`. Le planificateur ignore les critères d'échec lorsqu'une tâche du déploiement actuel se trouve dans l'état `RUNNING` et passe à l'étape suivante. Lorsque les tâches ne parviennent pas à atteindre l'état `RUNNING`, le disjoncteur de déploiement augmente d'un le nombre d'échecs. Lorsque le nombre d'échecs est égal au seuil, le déploiement est marqué comme étant `FAILED`.
2. Cette étape est entrée lorsqu'il y a une ou plusieurs tâches dans l'`RUNNING`état. Le disjoncteur de déploiement effectue une surveillance de l'état sur les ressources suivantes pour les tâches du déploiement actuel :
   - Equilibreurs de charge Elastic Load Balancing
   - AWS Cloud Map service
   - Surveillance de l'état des conteneurs Amazon ECS

   Lorsqu'une surveillance de l'état échoue pour la tâche, le disjoncteur de déploiement augmente d'un le nombre d'échecs. Lorsque le nombre d'échecs est égal au seuil, le déploiement est marqué comme étant `FAILED`.

Le tableau suivant fournit quelques exemples.

Nombre souhaité	Calculs	Seuil
1	`3 <= 0.5 * 1 => 200`	3 (la valeur calculée est inférieure au minimum)
25	`3 <= 0.5 * 25 => 200`	13 (la valeur est arrondie à la valeur supérieure)
400	`3 <= 0.5 * 400 => 200`	200
800	`3 <= 0.5 * 800 => 200`	200 (la valeur calculée est supérieure au maximum)

Par exemple, lorsque le seuil est de 3, le disjoncteur démarre avec le nombre de défaillances réglé à 0. Lorsqu'une tâche n'atteint pas `RUNNING` cet état, le disjoncteur de déploiement augmente le nombre d'échecs d'une unité. Lorsque le nombre d'échecs est égal à 3, le déploiement est marqué comme`FAILED`.

Pour obtenir des exemples supplémentaires sur l'utilisation de l'option de restauration, consultez Announcing Amazon ECS deployment circuit breaker (Annonce du disjoncteur de circuit de déploiement Amazon ECS).

Comment les CloudWatch alarmes détectent les échecs de déploiement d'Amazon ECS

Vous pouvez configurer Amazon ECS pour définir le déploiement comme un échec lorsqu'il détecte qu'une CloudWatch alarme spécifiée est passée dans `ALARM` cet état.

Vous pouvez éventuellement définir la configuration pour qu'elle revienne au dernier déploiement réussi en cas d'échec d'un déploiement.

L'`create-service` AWS CLI exemple suivant montre comment créer un service Linux lorsque les alarmes de déploiement sont utilisées avec l'option de restauration.

```
aws ecs create-service \
     --service-name MyService \
     --deployment-controller type=ECS \
     --desired-count 3 \
     --deployment-configuration
 "alarms={alarmNames=[alarm1Name,alarm2Name],enable=true,rollback=true}" \
```

```
    --task-definition sample-fargate:1 \
    --launch-type FARGATE \
    --platform-family LINUX \
    --platform-version 1.4.0 \
    --network-configuration
"awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
```

Tenez compte des points suivants lorsque vous utilisez la méthode CloudWatch des alarmes Amazon sur un service.

- Durée pendant laquelle les révisions du service bleu et vert sont exécutées simultanément après le déplacement du trafic de production. Amazon ECS calcule cette période en fonction de la configuration d'alarme associée au déploiement. Vous ne pouvez pas définir cette valeur.
- Le paramètre de demande `deploymentConfiguration` contient désormais le type de données `alarms`. Vous pouvez spécifier les noms des alarmes, si vous souhaitez utiliser la méthode et si vous souhaitez lancer une restauration lorsque les alarmes indiquent un échec de déploiement. Pour plus d'informations, consultez le CreateServicemanuel Amazon Elastic Container Service API Reference.
- La réponse `DescribeServices` donne un aperçu de l'état d'un déploiement, le `rolloutState` et `rolloutStateReason`. Lorsqu'un nouveau déploiement démarre, il commence à l'état `IN_PROGRESS`. Lorsque le service atteint un état stable et que la durée de l'intégration est terminée, l'état du déploiement passe à `COMPLETED`. Si le service n'arrive pas à atteindre un état stable et que l'alarme passe à l'état `ALARM`, le déploiement passe à un état `FAILED`. Un déploiement dans un état `FAILED` ne lancera aucune nouvelle tâche.
- Outre les événements de changement d'état de déploiement de service qu'Amazon ECS envoie pour les déploiements qui ont démarré et se sont terminés, Amazon ECS envoie également un événement lorsqu'un déploiement qui utilise des alarmes échoue. Ces événements fournissent des informations sur la raison pour laquelle un déploiement a échoué ou si un déploiement a été démarré en raison d'une restauration. Pour de plus amples informations, veuillez consulter Événements de changement d'état du déploiement du service Amazon ECS.
- Si un nouveau déploiement démarre parce qu'un déploiement précédent a échoué et que la restauration a été activée, le champ `reason` de l'événement de changement d'état de déploiement de service indique que le déploiement a démarré en raison d'une restauration.
- Si vous utilisez le disjoncteur de déploiement et les CloudWatch alarmes Amazon pour détecter les défaillances, l'un ou l'autre peut provoquer un échec du déploiement dès que les critères de l'une ou l'autre méthode sont remplis. Une restauration se produit lorsque vous utilisez l'option de restauration pour la méthode à l'origine de l'échec du déploiement.

- Les CloudWatch alarmes Amazon ne sont prises en charge que pour les services Amazon ECS qui utilisent le contrôleur de déploiement rolling update (`ECS`).
- Vous pouvez configurer cette option à l'aide de la console Amazon ECS ou du AWS CLI. Pour plus d'informations, consultez the section called “Créer un service à l'aide de paramètres définis” et create-service dans la Référence AWS Command Line Interface .
- Vous remarquerez peut-être que l'état du déploiement reste `IN_PROGRESS` pendant une durée prolongée. La raison en est qu'Amazon ECS ne modifie pas le statut tant qu'il n'a pas supprimé le déploiement actif, et cela ne se produit qu'après la durée de l'intégration. En fonction de votre configuration d'alarme, le déploiement peut sembler prendre plusieurs minutes de plus que lorsque vous n'utilisez pas d'alarmes (même si le nouvel ensemble de tâches principales est augmenté et l'ancien déploiement réduit). Si vous utilisez des CloudFormation délais d'attente, envisagez de les augmenter. Pour de plus amples informations, veuillez consulter Création de conditions d'attente dans un modèle dans le Guide de l'utilisateur AWS CloudFormation .
- Amazon ECS appelle `DescribeAlarms` pour sonder les alarmes. Les appels à prendre en `DescribeAlarms` compte dans les quotas de CloudWatch service associés à votre compte. Si d'autres AWS services appellent`DescribeAlarms`, Amazon ECS risque d'avoir un impact sur le fait d'interroger les alarmes. Par exemple, si un autre service effectue suffisamment d'`DescribeAlarms`appels pour atteindre le quota, ce service est limité et Amazon ECS est également limité et ne peut pas interroger les alarmes. Si une alarme est générée pendant la période de régulation, Amazon ECS risque de rater l'alarme et de ne pas procéder à la restauration. Cela n'a aucun autre impact sur le déploiement. Pour plus d'informations sur les quotas CloudWatch de service, consultez la section Quotas de CloudWatch service dans le Guide de CloudWatch l'utilisateur.
- Si une alarme est activée à l’état `ALARM` au début d'un déploiement, Amazon ECS ne surveille pas les alarmes pendant la durée de ce déploiement (il ignore la configuration des alarmes). Ce comportement permet de résoudre le cas où vous souhaitez démarrer un nouveau déploiement pour corriger un échec de déploiement initial.

Alarmes recommandées

Nous vous recommandons d'utiliser les métriques d'alarme suivantes :

- Si vous utilisez un Application Load Balancer, utilisez les métriques d'Application Load Balancer `HTTPCode_ELB_5XX_Count` et `HTTPCode_ELB_4XX_Count`. Ces métriques surveillent la présence de pics HTTP. Pour plus d'informations sur les métriques d'Application Load Balancer,

consultez CloudWatch les métriques de votre Application Load Balancer dans le Guide d'utilisation des Application Load Balancers.

- Si vous avez déjà une application, utilisez les métriques `CPUUtilization` et `MemoryUtilization`. Elles vérifient l'utilisation de l'UC et de la mémoire en pourcentage utilisées par le cluster ou le service. Pour de plus amples informations, veuillez consulter the section called “Considérations”.
- Si vous utilisez des Amazon Simple Queue Service files d'attente dans vos tâches, utilisez la métrique `ApproximateNumberOfMessagesNotVisible` Amazon SQS. Cette métrique vérifie le nombre de messages dans la file d'attente qui sont retardés et qui ne peuvent pas être lus immédiatement. Pour plus d'informations sur les métriques Amazon SQS, consultez la section Mesures disponibles CloudWatch pour Amazon SQS dans le manuel Amazon Simple Queue Service Developer Guide.

Hooks relatifs au cycle de vie pour les déploiements de services Amazon ECS

Lorsqu'un déploiement démarre, il passe par différentes étapes du cycle de vie. Ces étapes peuvent avoir un état tel que IN_PROGRESS ou être réussies. Vous pouvez utiliser les hooks du cycle de vie, qui sont des fonctions Lambda qu'Amazon ECS exécute en votre nom à des étapes du cycle de vie spécifiées. Les fonctions peuvent être l'une des suivantes :

- Une API asynchrone qui valide le bilan de santé en 15 minutes.
- Une API de sondage qui lance un autre processus asynchrone évaluant l'achèvement du cycle de vie du hook.

Une fois l'exécution de la fonction terminée, elle doit renvoyer un `hookStatus` pour que le déploiement se poursuive. Si a n'`hookStatus`est pas renvoyé ou si la fonction échoue, le déploiement est annulé. Les `hookStatus` valeurs sont les suivantes :

- `SUCCEEDED`- le déploiement se poursuit jusqu'à l'étape suivante du cycle de vie
- `FAILED`— le déploiement est reporté au dernier déploiement réussi.
- `IN_PROGRESS`— Amazon ECS exécute à nouveau la fonction après un court laps de temps. Par défaut, il s'agit d'un intervalle de 30 secondes, mais cette valeur est personnalisable en renvoyant un à `callBackDelaySeconds` côté de`hookStatus`.

L'exemple suivant montre comment renvoyer un `hookStatus` avec un délai de rappel personnalisé. Dans cet exemple, Amazon ECS réessaierait ce hook en 60 secondes au lieu des 30 secondes par défaut :

```
{
    "hookStatus": "IN_PROGRESS",
    "callBackDelaySeconds": 60
}
```

Lorsqu'un rollback se produit, Amazon ECS exécute les hooks du cycle de vie pour les étapes du cycle de vie suivantes :

- PRODUCTION_TRAFFIC_SHIFT
- TEST_TRAFFIC_SHIFT

Charges utiles du cycle de vie

Lorsque vous configurez des crochets de cycle de vie pour vos déploiements de services ECS, Amazon ECS invoque ces crochets à des étapes spécifiques du processus de déploiement. Chaque étape du cycle de vie fournit une charge utile JSON contenant des informations sur l'état actuel du déploiement. Ce document décrit la structure de charge utile pour chaque étape du cycle de vie.

Structure de charge utile commune

Toutes les charges utiles liées aux étapes du cycle de vie incluent les champs communs suivants :

- `serviceArn`- Le nom de ressource Amazon (ARN) du service.
- `targetServiceRevisionArn`- L'ARN de la révision du service cible en cours de déploiement.
- `testTrafficWeights`- Une carte de révision des services en fonction ARNs des pourcentages de pondération du trafic de test correspondants.
- `productionTrafficWeights`- Une carte de révision des services en fonction ARNs des pourcentages de pondération du trafic de production correspondants.

Charges utiles au stade du cycle de vie

SERVICE DE RÉCONCILIATION

Cette étape a lieu au début du processus de déploiement lorsque le service est en cours de réconciliation. Voici un exemple de charge utile pour cette étape du cycle de vie.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/01275892":
 100,
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/78652123": 0
  },
  "productionTrafficWeights": {
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/01275892":
 100,
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/78652123": 0
  }
}
```

Attentes à ce stade :

- L'ensemble de tâches principal est à une échelle de 0 %

PRÉ_SCALE_UP

Cette étape a lieu avant que les nouvelles tâches ne soient étendues. Voici un exemple de charge utile pour cette étape du cycle de vie.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {}
}
```

Attentes à ce stade :

- Les tâches de révision des services écologiques sont à une échelle de 0 %

POST_SCALE_UP

Cette étape intervient une fois que les nouvelles tâches ont été étendues et qu'elles sont saines. Voici un exemple de charge utile pour cette étape du cycle de vie.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {}
}
```

Attentes à ce stade :

- Les tâches de révision des services écologiques sont à une échelle de 100 %
- Les tâches liées à la révision des services écologiques sont saines

TEST_TRAFFIC_SHIFT

Cette étape se produit lorsque le trafic de test est transféré vers les tâches de révision des services écologiques.

Voici un exemple de charge utile pour cette étape du cycle de vie.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/01275892":
 100,
```

```
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/78652123": 0
  },
  "productionTrafficWeights": {}
}
```

Attentes à ce stade :

- Le trafic de test est en train de passer aux tâches de révision des services écologiques.

POST-TEST_TRAFFIC_SHIFT

Cette étape intervient une fois que le trafic de test a été entièrement transféré vers les nouvelles tâches.

Voici un exemple de charge utile pour cette étape du cycle de vie.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {}
}
```

Attentes à ce stade :

- 100 % du trafic de test a été affecté aux tâches de révision des services écologiques.

PRODUCTION_TRAFFIC_SHIFT

Cette étape se produit lorsque le trafic de production est transféré vers les tâches de révision des services écologiques.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
```

```
  "testTrafficWeights": {},
  "productionTrafficWeights": {
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/01275892":
 100,
    "arn:aws:ecs:us-west-2:1234567890:service-revision/myCluster/myService/78652123": 0
  }
}
```

Attentes à ce stade :

- Le trafic de production est en train de passer à la révision du service écologique.

POST-PRODUCTION TRAFFIC_SHIFT

Cette étape intervient une fois que le trafic de production a été entièrement transféré aux tâches de révision des services écologiques.

```
{
  "serviceArn": "arn:aws:ecs:us-west-2:1234567890:service/myCluster/myService",
  "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:1234567890:service-revision/
myCluster/myService/01275892",
  "testTrafficWeights": {},
  "productionTrafficWeights": {}
}
```

Attentes à ce stade :

- 100 % du trafic de production a été consacré aux tâches de révision des services écologiques.

Catégories d'étapes du cycle de vie

Les étapes du cycle de vie se répartissent en deux catégories :

1. Étapes d'appel uniques : ces étapes ne sont invoquées qu'une seule fois lors du déploiement d'un service :
   - PRÉ_SCALE_UP
   - POST_SCALE_UP

- POST-TEST_TRAFFIC_SHIFT
- POST-PRODUCTION TRAFFIC_SHIFT

2. Étapes d'appel récurrentes : ces étapes peuvent être invoquées plusieurs fois au cours du déploiement d'un service, par exemple lors d'une opération de restauration :
   - TEST_TRAFFIC_SHIFT
   - PRODUCTION_TRAFFIC_SHIFT

État du déploiement pendant les phases de cycle de vie

Pendant que les hooks du cycle de vie sont en cours d'exécution, l'état du déploiement s'appliquera `IN_PROGRESS` à toutes les étapes du cycle de vie.

Étape du cycle de vie	État du déploiement
SERVICE DE RÉCONCILIATION	EN_COURS
PRÉ_SCALE_UP	EN_COURS
POST_SCALE_UP	EN_COURS
TEST_TRAFFIC_SHIFT	EN_COURS
POST-TEST_TRAFFIC_SHIFT	EN_COURS
PRODUCTION_TRAFFIC_SHIFT	EN_COURS
POST-PRODUCTION TRAFFIC_SHIFT	EN_COURS

Arrêt des déploiements de services Amazon ECS

Vous pouvez arrêter manuellement un déploiement lorsqu'un déploiement défaillant n'a pas été détecté par le disjoncteur ou les CloudWatch alarmes. Les types d'arrêt suivants sont disponibles :

- Annulation : cette option ramène le déploiement du service à la version précédente du service.

  Vous pouvez utiliser cette option même si vous n'avez pas configuré le déploiement du service pour l'option de restauration.

Vous pouvez arrêter un déploiement qui se trouve dans l'un des états suivants. Pour plus d'informations sur les états de déploiement des services, consultezAfficher l'historique des services à l'aide des déploiements de services Amazon ECS.

- EN ATTENTE - Le déploiement du service passe à l'état ROLLBACK_REQUESTED, puis l'opération de restauration commence.
- IN_PROGRESS - Le déploiement du service passe à l'état ROLLBACK_REQUESTED, puis l'opération de restauration commence.
- STOP_REQUESTED - Le déploiement du service continue de s'arrêter.
- ROLLBACK_REQUESTED - Le déploiement du service poursuit l'opération de restauration.
- ROLLBACK_IN_PROGRESS - Le déploiement du service poursuit l'opération de restauration.

Procédure

Avant de commencer, configurez les autorisations requises pour visualiser les déploiements de services. Pour de plus amples informations, veuillez consulter Autorisations requises pour consulter les déploiements de services Amazon ECS.

Amazon ECS Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page des détails du service s'affiche.
4. Sur la page des détails du service, sélectionnez Déploiements.

   La page des déploiements s'affiche.
5. Sous Déploiement en cours, choisissez Roll back. Ensuite, dans la fenêtre de confirmation, choisissez Roll back.

AWS CLI

1. Exécutez `list-service-deployments` pour récupérer l'ARN de déploiement du service.

   Remplacez le *`user-input`* par vos valeurs.

```
aws ecs list-service-deployments --cluster cluster-name --service service-name
```

Notez le `serviceDeploymentArn` nom du déploiement que vous souhaitez arrêter.

```
{
    "serviceDeployments": [
        {
            "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-
deployment/cluster-name/service-name/NCWGC2ZR-taawPAYrIaU5",
            "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/cluster-
name/service-name",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/cluster-
name",
            "targetServiceRevisionArn": "arn:aws:ecs:us-
west-2:123456789012:service-revision/cluster-name/service-
name/4980306466373577095",
            "status": "SUCCESSFUL"
        }
    ]
}
```

2. Exécutez `stop-service-deployments`. Utilisez celui `serviceDeploymentArn` qui a été renvoyé`list-service-deployments`.

   Remplacez le *`user-input`* par vos valeurs.

```
aws ecs stop-service-deployment --service-deployment-arn
 arn:aws:ecs:region:123456789012:service-deployment/cluster-name/service-
name/NCWGC2ZR-taawPAYrIaU5 --stop-type ROLLBACK
```

Étapes suivantes

Décidez quelles modifications doivent être apportées au service, puis mettez-le à jour. Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.

Déployez les services Amazon ECS en remplaçant les tâches

Lorsque vous créez un service qui utilise le type de déploiement rolling update (`ECS`), le planificateur de services Amazon ECS remplace les tâches en cours d'exécution par de nouvelles tâches. Le

nombre de tâches ajoutées au service ou supprimées du service par Amazon ECS lors d'une mise à jour propagée est contrôlé par la configuration de déploiement du service.

Amazon ECS utilise les paramètres suivants pour déterminer le nombre de tâches :

- `minimumHealthyPercent` représente la limite inférieure du nombre de tâches qui doivent être exécutées pour un service pendant un déploiement ou lorsqu'une instance de conteneur est drainée, en pourcentage du nombre souhaité de tâches pour le service. Cette valeur est arrondie à la valeur supérieure. Par exemple, si le pourcentage minimum d'instances saines est `50` et que le nombre de tâches souhaité est de quatre, le planificateur peut arrêter deux tâches existantes avant de démarrer deux nouvelles tâches. De même, si le pourcentage de santé minimum est de 75 % et que le nombre de tâches souhaité est de deux, le planificateur ne peut pas arrêter de tâche car la valeur résultante est également de deux.

  Si les tâches ne fonctionnent plus correctement, le planificateur de services Amazon ECS commence d'abord par les tâches de remplacement et assure la maintenance des tâches jusqu'à ce que les `minimumHealthyPercent` tâches de remplacement redeviennent saines. Au fur et à mesure que les tâches de remplacement démarrent et deviennent saines, les tâches malsaines seront progressivement arrêtées.
- `maximumPercent` représente la limite supérieure du nombre de tâches qui doivent être exécutées pour un service pendant un déploiement ou lorsqu'une instance de conteneur est drainée, en pourcentage du nombre souhaité de tâches pour un service. Cette valeur est arrondie à la valeur inférieure. Par exemple, si le pourcentage maximal est `200` et que le nombre de tâches souhaitées est quatre, le planificateur peut démarrer quatre nouvelles tâches avant d'arrêter quatre tâches existantes. De même, si le pourcentage maximal est `125` et que le nombre de tâches souhaité est de trois, le planificateur ne peut pas démarrer de tâche, car la valeur résultante est également de trois.

> **Important**
>
> Lorsque vous définissez un pourcentage minimum ou maximal d'instances saines, vous devez vous assurer que le planificateur peut arrêter ou démarrer au moins une tâche lorsqu'un déploiement est déclenché. Si votre service a un déploiement bloqué en raison d'une configuration de déploiement non valide, un message d'événement de service est envoyé. Pour de plus amples informations, veuillez consulter service (*`service-name`*) n'a pas pu arrêter ou démarrer des tâches pendant un déploiement en raison de la

configuration du déploiement du service. Mettez à jour la valeur minimumHealthyPercent ou MaximumPercent et réessayez..

Les déploiements progressifs ont deux méthodes qui permettent d'identifier rapidement l'échec d'un déploiement de service :

- the section called “ Comment le disjoncteur de déploiement détecte les défaillances”
- the section called “Comment les CloudWatch alarmes détectent les échecs de déploiement”

Les méthodes peuvent être utilisées séparément ou conjointement. Lorsque les deux méthodes sont utilisées, le déploiement est défini comme ayant échoué dès que les critères d'échec de l'une ou l'autre méthode d'échec sont remplis.

Suivez les instructions suivantes pour déterminer quelle méthode utiliser :

- Disjoncteur : utilisez cette méthode lorsque vous souhaitez arrêter un déploiement car les tâches ne peuvent pas démarrer.
- CloudWatch alarmes - Utilisez cette méthode lorsque vous souhaitez arrêter un déploiement en fonction des métriques de l'application.

Les deux méthodes permettent de revenir à la révision du service précédente.

Résolution de l'image du conteneur

Par défaut, Amazon ECS convertit les balises d'image de conteneur spécifiées dans la définition de tâche en résumés d'images de conteneurs. Si vous créez un service qui exécute et gère une seule tâche, cette tâche est utilisée pour établir des résumés d'images pour les conteneurs de la tâche. Si vous créez un service qui exécute et gère plusieurs tâches, la première tâche démarrée par le planificateur de services lors du déploiement est utilisée pour établir les résumés d'images pour les conteneurs contenus dans les tâches.

Si au moins trois tentatives d'établissement des résumés d'images du conteneur échouent, le déploiement se poursuit sans résolution de condensé d'image. Si le disjoncteur de déploiement est activé, le déploiement échoue en outre et est annulé.

Une fois les résumés des images du conteneur établis, Amazon ECS les utilise pour démarrer toutes les autres tâches souhaitées et pour toute future mise à jour de service. Ainsi, toutes les tâches d'un

service exécutent toujours des images de conteneur identiques, ce qui garantit la cohérence des versions de votre logiciel.

Vous pouvez configurer ce comportement pour chaque conteneur de votre tâche en utilisant le `versionConsistency` paramètre figurant dans la définition du conteneur. Pour de plus amples informations, veuillez consulter `versionConsistency`.

> **Note**
>
> - Les versions d'Amazon ECS Agent inférieures à `1.31.0` ne prennent pas en charge la résolution du condensé d'image. Versions `1.31.0` d'agent prenant en `1.69.0` charge la résolution de condensé d'images uniquement pour les images transférées vers les référentiels Amazon ECR. Les versions de l'agent `1.70.0` ou supérieures prennent en charge la résolution de résumé d'image pour toutes les images.
> - La version minimale de la plate-forme Fargate Linux pour la résolution du condensé d'image est. `1.3.0` La version minimale de la plate-forme Fargate Windows pour la résolution du condensé d'image est. `1.0.0`
> - Amazon ECS ne capture pas les résumés des conteneurs annexes gérés par Amazon ECS, tels que l'agent de GuardDuty sécurité Amazon ou le proxy Service Connect.
> - Pour réduire le temps de latence potentiel associé à la résolution des images de conteneur dans les services comportant plusieurs tâches, exécutez la version `1.83.0` ou supérieure de l'agent Amazon ECS sur les instances de EC2 conteneur. Pour éviter toute latence potentielle, spécifiez des résumés d'images de conteneur dans votre définition de tâche.
> - Si vous créez un service avec un nombre de tâches souhaité égal à zéro, Amazon ECS ne peut pas établir de résumés de conteneurs tant que vous n'avez pas déclenché un autre déploiement du service avec un nombre de tâches souhaité supérieur à zéro.
> - Pour établir des résumés d'images mis à jour, vous pouvez forcer un nouveau déploiement. Les résumés mis à jour seront utilisés pour démarrer de nouvelles tâches et n'affecteront pas les tâches déjà en cours. Pour plus d'informations sur le forçage de nouveaux déploiements, consultez forceNewDeploymentla référence des API Amazon ECS.
> - Lorsque vous faites appel à des fournisseurs de EC2 capacité, si la capacité est insuffisante pour démarrer une tâche lors du déploiement initial, la cohérence des versions logicielles peut échouer. Pour garantir la cohérence des versions même lorsque la capacité est limitée, définissez-la explicitement `versionConsistency: true` dans la configuration de votre conteneur de définition de tâches plutôt que de vous fier au

comportement par défaut. Cela oblige Amazon ECS à attendre que la capacité soit disponible avant de procéder au déploiement.

Bonnes pratiques relatives aux paramètres de service Amazon ECS

Pour éviter toute interruption de service des applications, le processus de déploiement est le suivant :

1. Démarrez les nouveaux conteneurs d'applications tout en maintenant les conteneurs existants en activité.
2. Vérifiez que les nouveaux contenants sont sains.
3. Arrêtez les vieux conteneurs.

En fonction de la configuration de votre déploiement et de la quantité d'espace libre non réservé dans votre cluster, plusieurs cycles peuvent être nécessaires pour remplacer toutes les anciennes tâches par de nouvelles tâches.

Il existe deux options de configuration de service que vous pouvez utiliser pour modifier le numéro :

- `minimumHealthyPercent`: 100 % (par défaut)

  Limite inférieure du nombre de tâches de votre service qui doivent rester `RUNNING` inchangées pendant un déploiement. Il s'agit d'un pourcentage de l'entier `desiredCount` arrondi à l'entier supérieur le plus proche. Ce paramètre vous permet de déployer sans utiliser de capacité de cluster supplémentaire.
- `maximumPercent`: 200 % (par défaut)

  Limite supérieure du nombre de tâches de votre service autorisées dans l'`PENDING`état `RUNNING` ou lors d'un déploiement. Il s'agit d'un pourcentage de l'entier `desiredCount` arrondi à l'entier inférieur le plus proche.

Exemple : options de configuration par défaut

Prenons l'exemple du service suivant qui comporte six tâches, déployé dans un cluster pouvant accueillir huit tâches au total. Les options de configuration du service par défaut n'autorisent pas le déploiement à moins de 100 % des six tâches souhaitées.

Le processus de déploiement est le suivant :

1. L'objectif est de remplacer les six tâches.
2. Le planificateur lance deux nouvelles tâches car les paramètres par défaut exigent que six tâches soient en cours d'exécution.

   Il existe désormais six tâches existantes et deux nouvelles tâches.
3. Le planificateur arrête deux des tâches existantes.

   Il existe désormais quatre tâches existantes et deux nouvelles.
4. Le planificateur lance deux nouvelles tâches supplémentaires.

   Il existe désormais quatre tâches existantes et quatre nouvelles tâches.
5. Le planificateur arrête deux des tâches existantes.

   Il existe désormais deux tâches existantes et quatre nouvelles.
6. Le planificateur lance deux nouvelles tâches supplémentaires.

   Il y a maintenant deux tâches existantes et six nouvelles tâches
7. Le planificateur arrête les deux dernières tâches existantes.

   Il existe désormais six nouvelles tâches.

Dans l'exemple ci-dessus, si vous utilisez les valeurs par défaut pour les options, il y a une attente de 2,5 minutes pour chaque nouvelle tâche qui démarre. En outre, l'équilibreur de charge devra peut-être attendre 5 minutes pour que l'ancienne tâche s'arrête.

Exemple : Modifier **`minimumHealthyPercent`**

Vous pouvez accélérer le déploiement en définissant la `minimumHealthyPercent` valeur sur 50 %.

Prenons l'exemple du service suivant qui comporte six tâches, déployé dans un cluster pouvant accueillir huit tâches au total. Le processus de déploiement est le suivant :

1. L'objectif est de remplacer six tâches.
2. Le planificateur arrête trois des tâches existantes.

   Trois tâches correspondant à cette `minimumHealthyPercent` valeur sont toujours en cours d'exécution.
3. Le planificateur lance cinq nouvelles tâches.

Il existe trois tâches existantes et cinq nouvelles tâches.

4. Le planificateur arrête les trois tâches existantes restantes.

   Il y a cinq nouvelles tâches

5. Le planificateur lance les dernières nouvelles tâches.

   Il existe six nouvelles tâches.

Exemple : modifier l'espace libre du cluster

Vous pouvez également ajouter de l'espace libre afin de pouvoir exécuter des tâches supplémentaires.

Prenons l'exemple du service suivant qui comporte six tâches, déployées dans un cluster pouvant accueillir dix tâches au total. Le processus de déploiement est le suivant :

1. L'objectif est de remplacer les tâches existantes.
2. Le planificateur arrête trois des tâches existantes,

   Trois tâches existent déjà.

3. Le planificateur lance six nouvelles tâches.

   Il y a les tâches existantes et six nouvelles tâches

4. Le planificateur arrête les trois tâches existantes.

   Il existe six nouvelles tâches.

Recommandations

Utilisez les valeurs suivantes pour les options de configuration du service lorsque vos tâches sont inactives pendant un certain temps et que leur taux d'utilisation n'est pas élevé.

- `minimumHealthyPercent`: 50 %
- `maximumPercent`: 200 %

Création d'un déploiement de mises à jour continues Amazon ECS

Créez un service pour exécuter et gérer un nombre spécifié d'instances d'une définition de tâche simultanément dans un cluster. Si l'une de vos tâches échoue ou s'arrête, le planificateur de service d'Amazon ECS service lance une autre instance de votre définition de tâche pour la remplacer. Cela permet de maintenir le nombre de tâches souhaité dans le service.

Définissez les paramètres de configuration suivants avant de créer un service :

- Deux options de calcul permettent de distribuer vos tâches.
  - Une capacity provider strategy (stratégie de fournisseurs de capacités) permet à Amazon ECS de distribuer vos tâches avec un ou plusieurs fournisseurs de capacité.
  - Un type de lancement permet à Amazon ECS de lancer nos tâches directement sur Fargate ou sur EC2 les instances enregistrées dans vos clusters.
- Les définitions de tâches qui utilisent le mode réseau `awsvpc` ou les services configurés pour utiliser un équilibreur de charge doivent avoir une configuration réseau. Par défaut, la console sélectionne l'Amazon VPC par défaut ainsi que tous ses sous-réseaux et son groupe de sécurité par défaut.
- La stratégie de placement, La stratégie de placement des tâches par défaut répartit les tâches de manière uniforme entre les zones de disponibilité.

  Nous vous recommandons d'utiliser le rééquilibrage des zones de disponibilité pour garantir la haute disponibilité de votre service. Pour de plus amples informations, veuillez consulter Équilibrer un service Amazon ECS entre les zones de disponibilité.
- Lorsque vous utilisez Launch Type (Type de lancement) pour le déploiement de votre service, le service démarre par défaut dans les sous-réseaux de votre VPC de cluster.
- Pour la stratégie des fournisseurs de capacités, la console sélectionne une option de calcul par défaut. Voici l'ordre utilisé par la console pour sélectionner une valeur par défaut :
  - Si votre cluster a une stratégie de fournisseur de capacité définie par défaut, elle est sélectionnée.
  - Si aucune stratégie de fournisseur de capacité par défaut n'est définie pour votre cluster mais que les fournisseurs de capacité Fargate y ont été ajoutés, une stratégie de fournisseur de capacité personnalisée utilisant `FARGATE` le fournisseur de capacité est sélectionnée.
  - Si aucune stratégie de fournisseur de capacité par défaut n'est définie pour votre cluster, mais qu'un ou plusieurs fournisseurs de capacité de groupe Auto Scaling y ont été ajoutés, l'option Use custom (Advanced) est sélectionnée et vous devez définir manuellement la stratégie.

  - Si votre cluster n'a pas de stratégie de fournisseur de capacité par défaut définie et qu'aucun fournisseur de capacité n'est ajouté au cluster, le type de lancement Fargate est sélectionné.
- Les options par défaut de détection des défaillances de déploiement consistent à utiliser l'option de disjoncteur de déploiement Amazon ECS avec l'option Rollback en cas d'échec.

  Pour de plus amples informations, veuillez consulter Comment le disjoncteur de déploiement Amazon ECS détecte les défaillances.
- Si vous souhaitez utiliser l'option de blue/green déploiement, déterminez le mode de CodeDeploy déplacement des applications. Les options suivantes sont disponibles :
  - CodeDeployDefault. ECSAllAtOnce: Transfère immédiatement tout le trafic vers le conteneur Amazon ECS mis à jour
  - CodeDeployDefault. ECSLinear10 PercentEvery 1Minutes : déplace 10 % du trafic toutes les minutes jusqu'à ce que tout le trafic soit transféré.
  - CodeDeployDefault. ECSLinear10 à PercentEvery 3 minutes : déplace 10 % du trafic toutes les 3 minutes jusqu'à ce que tout le trafic soit transféré.
  - CodeDeployDefault. ECSCanary10 % 5 minutes : déplace 10 % du trafic dès le premier incrément. Les 90 % restants sont déployés 5 minutes plus tard.
  - CodeDeployDefault. ECSCanary10 % 15 minutes : déplace 10 % du trafic dès le premier incrément. Les 90 % restants sont déployés 15 minutes plus tard.
- Décidez si vous souhaitez qu'Amazon ECS augmente ou diminue automatiquement le nombre de tâches souhaité dans votre service. Pour plus d'informations, consultez Faites évoluer automatiquement votre service Amazon ECS.
- Si vous avez besoin d'une application pour vous connecter à d'autres applications qui s'exécutent dans Amazon ECS, déterminez l'option adaptée à votre architecture. Pour de plus amples informations, veuillez consulter Interconnectez les services Amazon ECS.
- Lorsque vous créez un service qui utilise le disjoncteur Amazon ECS, Amazon ECS crée un déploiement et une révision du service. Ces ressources vous permettent de consulter des informations détaillées sur l'historique des services. Pour de plus amples informations, veuillez consulter Afficher l'historique des services à l'aide des déploiements de services Amazon ECS.

  Pour plus d'informations sur la création d'un service à l'aide du AWS CLI, reportez-vous create-serviceà la section AWS Command Line Interface Référence.

  Pour plus d'informations sur la création d'un service à l'aide de AWS CloudFormation, reportez-vous AWS::ECS::Serviceau Guide de AWS CloudFormation l'utilisateur.

Création d'un service avec les options par défaut

Vous pouvez utiliser la console pour créer et déployer rapidement un service. Le service a la configuration suivante :

- Déploie dans le VPC et les sous-réseaux associés à votre cluster
- Déploie une tâche
- Utilise le déploiement continu
- Utilise la stratégie de fournisseur de capacité avec votre fournisseur de capacité par défaut
- Utilise le disjoncteur de déploiement pour détecter les défaillances et définit l'option permettant d'annuler automatiquement le déploiement en cas de problème

Pour déployer un service à l'aide des paramètres par défaut, procédez comme suit.

Pour créer un service (console Amazon ECS)

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la page de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster dans lequel créer le service.
4. Sous l'onglet Services choisissez Create (Créer).

   La page Créer un service apparaît.
5. Sous Détails du service, procédez comme suit :

   a. Pour Définition de tâche, entrez la famille de définitions de tâches et la révision à utiliser.

   b. Pour Service name (Nom du service), saisissez un nom pour votre service.
6. Sous Configuration du déploiement, procédez comme suit :

   - Pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer et à conserver dans le service.
7. (Facultatif) Pour vous aider à identifier votre service et vos tâches, développez Tags (balises), puis configurez vos balises.

   Pour qu'Amazon ECS étiquette automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises de définition des tâches, sélectionnez Turn on Amazon ECS managed tags (Activer les balises gérées par Amazon ECS), puis sélectionnez Task definitions (Définitions de tâches).

Pour qu'Amazon ECS étiquette automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises d'un service, sélectionnez Turn on Amazon ECS managed tags (Activer les balises gérées par Amazon ECS), puis sélectionnez Service.

Ajoutez ou supprimez une balise.

- [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :
  - Pour Clé, saisissez le nom de la clé.
  - Pour Valeur, saisissez la valeur de clé.
- [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.

Créer un service à l'aide de paramètres définis

Pour créer un service à l'aide de paramètres définis, procédez comme suit.

Pour créer un service (console Amazon ECS)

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Déterminez la ressource à partir de laquelle vous lancez le service.

Pour démarrer un service à partir de	Étapes	
Clusters	a. Sur la page Clusters, sélectionnez le cluster dans lequel créer le service. b. Sous l'onglet Services choisissez Create (Créer).	
Définition de tâche	a. Sur la page Définitions de tâches, sélectionnez le bouton d'option situé à côté de la définition de tâche.	

Pour démarrer un service à partir de	Étapes	
	b. Dans le menu Déployer, choisissez Create service.	

La page Créer un service apparaît.

3. Sous Détails du service, procédez comme suit :

   a. Pour Définition de tâche, entrez la définition de tâche à utiliser. Ensuite, pour Révision, choisissez la révision à utiliser.

   b. Pour Service name (Nom du service), saisissez un nom pour votre service.

4. Pour Cluster existant, choisissez le cluster.

   Choisissez Créer un cluster pour exécuter la tâche sur un nouveau cluster

5. Choisissez la manière dont vos tâches sont réparties dans l'infrastructure de votre cluster. Sous Configuration informatique, choisissez votre option.

Option de calcul	Étapes	
Stratégie de fournisseur de capacité	a. Sous Options de calcul, choisissez Stratégie de fournisseur de capacité. b. Choisissez une stratégie : • Pour utiliser une stratégie de fournisseur de capacité, choisisse z Use cluster default (Utiliser la stratégie par défaut du cluster). • Si votre cluster ne possède pas de stratégie de fournisse ur de capacité par défaut ou si vous	

Option de calcul	Étapes	
	souhaitez utiliser une stratégie personnalisée, sélectionnez Utiliser une stratégie personnalisée ou Ajouter une stratégie de fournisseur de capacité et définissez votre stratégie de fournisseur de capacité personnalisée en spécifiant une Base, un Fournisseur de capacité et un Poids. **Note** Pour utiliser un fournisseur de capacité dans une stratégie, le fournisseur de capacité doit être associé au cluster.	

Option de calcul	Étapes	
Type de lancement	a. Dans Compute options (Options de calcul), sélectionnez Launch type (Type de lancement). b. Pour Launch type (Type de lancement), choisissez un type de lancement. c. (Facultatif) Lorsque le type de lancement Fargate est spécifié, pour Platform version (Version de plateforme), spécifiez la version de plateform e à utiliser. Si vous ne spécifiez aucune version de plateforme, la version `LATEST` est utilisée.	

6. Sous Configuration du déploiement, procédez comme suit :

   a. Pour Service type (Type de service), choisissez la stratégie de planification du service.

      - Pour que le planificateur déploie exactement une tâche sur chaque instance de conteneur active qui répond à toutes les contraintes de placement des tâches spécifiées dans votre cluster, choisissez Daemon (Démon).
      - Pour que. le planificateur place et gère le nombre de tâches souhaité dans votre cluster, choisissez Replica.

   b. Si vous choisissez Replica pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer et à conserver dans le service.

   c. Si vous avez choisi Replica, pour qu'Amazon ECS surveille la répartition des tâches entre les zones de disponibilité et les redistribue en cas de déséquilibre, sous Rééquilibrage du service des zones de disponibilité, sélectionnez Rééquilibrage du service des zones de disponibilité.

d. Pour la période de grâce du bilan de santé, entrez la durée (en secondes) pendant laquelle le planificateur de services ignore les contrôles d'intégrité d'Elastic Load Balancing, de VPC Lattice et de conteneurs malsains après le premier démarrage d'une tâche. Si vous ne spécifiez aucune valeur de période de grâce pour le bilan de santé, la valeur par défaut de 0 est utilisée.

e. Déterminez le type de déploiement de votre service. Développez les options de déploiement, puis spécifiez les paramètres suivants.

Type de déploiement	Étapes	
La mise à jour propagée	a. Pour Min running tasks (Minimum de tâches en cours d'exécuti on), saisissez la limite inférieure pour le nombre de tâches du service qui doivent rester à l'état `RUNNING` lors d'un déploiement, en tant que pourcentage de nombre souhaité de tâches (arrondi au nombre entier supérieur le plus proche). Pour plus d'informations, consultez Deployment configuration (Configur ation de déploiement). b. Pour Max running tasks (Maximum de tâches en cours d'exécuti on), saisissez la limite supérieure pour le nombre de tâches du service qui peuvent rester à l'état `RUNNING` ou `PENDING` lors d'un déploiement, en tant que pourcentage de nombre souhaité de tâches (arrondi au nombre entier inférieur le plus proche).	

Type de déploiement	Étapes	
Déploiement bleu/vert	a. Pour la configuration du déploiement, choisissez le mode d' CodeDeploy acheminement du trafic de production vers votre ensemble de tâches de remplacement lors d'un déploiement. b. Dans Rôle de service pour CodeDeploy, choisissez le rôle IAM que le service utilise pour envoyer des demandes d'API à des personnes autorisées Services AWS.	

f. Pour configurer la manière dont Amazon ECS détecte et gère les échecs de déploiement, développez Deployment failure detection (Détection des échecs de déploiement), puis choisissez vos options.

   i. Pour arrêter un déploiement lorsque les tâches ne peuvent pas démarrer, sélectionnez Use the Amazon ECS deployment circuit breaker (Utiliser le disjoncteur de déploiement Amazon ECS).

      Pour que le logiciel annule automatiquement le déploiement au dernier état de déploiement terminé lorsque le disjoncteur de déploiement met le déploiement en état d'échec, sélectionnez Annulation en cas d'échec.

   ii. Pour arrêter un déploiement en fonction des métriques de l'application, sélectionnez Utiliser une ou plusieurs CloudWatch alarmes. Ensuite, à partir du nom de l'CloudWatch alarme, choisissez les alarmes. Pour créer une nouvelle alarme, rendez-vous sur la CloudWatch console.

Pour que le logiciel annule automatiquement le déploiement au dernier état de déploiement terminé lorsqu'une CloudWatch alarme indique que le déploiement a échoué, sélectionnez Annulation en cas d'échec.

7. Si votre définition de tâche utilise le mode `awsvpc` réseau, vous pouvez spécifier une configuration réseau personnalisée, étendre la mise en réseau, puis effectuer les opérations suivantes pour.

   a. Pour VPC, sélectionnez le VPC à utiliser.

   b. Pour Subnets (Sous-réseaux), sélectionnez un ou plusieurs sous-réseaux du VPC que le planificateur de tâches prend en compte lors du placement de vos tâches.

   c. Pour Security group (Groupe de sécurité), vous pouvez sélectionner un groupe de sécurité existant ou en créer un nouveau. Pour utiliser un groupe de sécurité existant, sélectionnez-le et passez à l'étape suivante. Pour créer un nouveau groupe de sécurité, sélectionnez Create a new security group (Créer un nouveau groupe de sécurité). Vous devez spécifier un nom de groupe de sécurité et une description, puis ajouter une ou plusieurs règles entrantes pour le groupe de sécurité.

   d. Pour Public IP (Adresse IP publique), indiquez si vous voulez attribuer automatiquement une adresse IP publique à l'interface réseau Elastic (ENI) de la tâche.

      AWS Fargate les tâches peuvent se voir attribuer une adresse IP publique lorsqu'elles sont exécutées dans un sous-réseau public afin qu'elles disposent d'une route vers Internet. EC2 il n'est pas possible d'attribuer une adresse IP publique aux tâches à l'aide de ce champ. Pour plus d'informations, consultez les options de mise en réseau des tâches Amazon ECS pour le type de lancement Fargate et Allocation d'une interface réseau pour une tâche Amazon ECS.

8. (Facultatif) Pour interconnecter votre service à l'aide de Service Connect, étendez Service Connect, puis spécifiez les éléments suivants :

   a. Sélectionnez Activer Service Connect.

   b. Sous Service Connect configuration (Configuration de Service Connect), spécifiez le mode client.

      - Si votre service exécute une application client réseau qui doit uniquement se connecter aux autres services de l'espace de noms, choisissez Client side only.

- Si votre service exécute une application réseau ou un service Web et doit fournir des points de terminaison pour ce service, et se connecte à d'autres services dans l'espace de noms, choisissez Client and server (Client et serveur).

c. Pour utiliser un espace de noms qui n'est pas l'espace de noms de cluster par défaut, dans Namespace (Espace de noms), choisissez l'espace de noms du service.

d. (Facultatif) Spécifiez une configuration de journal. Sélectionnez Utiliser la collecte de journaux. L'option par défaut envoie les journaux du conteneur à CloudWatch Logs. Les autres options du pilote de journal sont configurées à l'aide de AWS FireLens. Pour de plus amples informations, veuillez consulter Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

   Voici une description plus détaillée de chaque destination de journal de conteneur.

   - Amazon CloudWatch — Configurez la tâche pour envoyer les journaux des conteneurs à CloudWatch Logs. Les options du pilote de journal par défaut sont fournies, ce qui permet de créer un groupe de CloudWatch journaux en votre nom. Pour spécifier un autre nom de groupe de journaux, modifiez les valeurs des options de pilote.
   - Amazon Data Firehose : configurez la tâche pour envoyer les journaux des conteneurs à Firehose. Les options du pilote de journal par défaut sont fournies, qui envoient les journaux à un flux de diffusion Firehose. Pour spécifier un autre nom de flux de diffusion, modifiez les valeurs des options de pilote.
   - Amazon Kinesis Data Streams : configurez la tâche pour envoyer les journaux des conteneurs à Kinesis Data Streams. Les options du pilote de journal par défaut sont fournies, qui envoient les journaux vers un flux Kinesis Data Streams. Pour spécifier un autre nom de flux, modifiez les valeurs des options de pilote.
   - Amazon OpenSearch Service — Configurez la tâche pour envoyer les journaux des conteneurs vers un domaine OpenSearch de service. Les options de pilote de journal doivent être fournies.
   - Amazon S3 — Configurez la tâche pour envoyer les journaux des conteneurs vers un compartiment Amazon S3. Les options du pilote de journal par défaut sont fournies, mais vous devez spécifier un nom de compartiment Amazon S3 valide.

9. (Facultatif) Pour interconnecter votre service à l'aide de Service Discovery, développez Service Discovery, puis procédez comme suit.

   a. Sélectionnez Utiliser la découverte de services.

b. Pour utiliser un nouvel espace de noms, choisissez Créer un nouvel espace de noms sous Configurer l'espace de noms, puis fournissez un nom et une description de l'espace de noms. Pour utiliser un espace de noms existant, choisissez Sélectionner un espace de noms existant, puis choisissez l'espace de noms que vous souhaitez utiliser.

c. Fournissez des informations sur le service Service Discovery, telles que le nom et la description du service.

d. Pour qu'Amazon ECS effectue des contrôles de santé périodiques au niveau du conteneur, sélectionnez Activer la propagation de l'état des tâches Amazon ECS.

e. Pour DNS record type (Type de registre DNS), sélectionnez le type de registre DNS à créer pour votre service. Amazon ECS Service Discovery ne prend en charge que les enregistrements A et SRV, selon le mode réseau spécifié par votre définition de tâche. Pour de plus amples informations sur ces types de registres, veuillez consulter Supported DNS Record Types (Types de registres DNS pris en charge) dans le Guide du développeur Amazon Route 53.

   - Si la définition de tâche spécifiée par votre tâche de service utilise le mode réseau `bridge` ou `host`, seuls les enregistrements de type SRV sont pris en charge. Choisissez une combinaison de nom et de port de conteneur à associer au registre.
   - Si la définition de tâche spécifiée par votre tâche de service utilise le mode réseau `awsvpc`, sélectionnez le type d'enregistrement A ou SRV. Si vous choisissez A, passez à l'étape suivante. Si vous choisissez SRV, spécifiez le port sur lequel le service est accessible ou une combinaison de nom et de port de conteneur à associer à l'enregistrement.

   Pour Durée de vie, saisissez la durée en secondes pendant laquelle un jeu d'enregistrements est conservé en cache par les résolveurs DNS et les navigateurs Web.

10. (Facultatif) Pour interconnecter votre service à l'aide de VPC Lattice, xxpand VPC Lattice, puis procédez comme suit :

    a. Sélectionnez Activer le réseau VPC

    b. Pour Rôle d'infrastructure, choisissez le rôle d'infrastructure.

       Si vous n'avez pas créé de rôle, choisissez Créer un rôle d'infrastructure.

    c. Sous Groupes cibles, sélectionnez le ou les groupes cibles. Vous devez choisir au moins un groupe cible et cinq au maximum. Choisissez Ajouter un groupe cible pour ajouter des

groupes cibles supplémentaires. Choisissez le nom du port, le protocole et le port pour chaque groupe cible que vous avez choisi.

Pour supprimer un groupe cible, choisissez Supprimer.

> **Note**
>
> - Si vous souhaitez ajouter des groupes cibles existants, vous devez utiliser le AWS CLI. Pour obtenir des instructions sur la façon d'ajouter des groupes cibles à l'aide de AWS CLI, voir register-targets dans la AWS Command Line Interface référence.
> - Bien qu'un service VPC Lattice puisse avoir plusieurs groupes cibles, chaque groupe cible ne peut être ajouté qu'à un seul service.

d. Pour terminer la configuration de VPC Lattice, en incluant vos nouveaux groupes cibles dans l'action par défaut de l'écouteur ou dans les règles d'un service VPC Lattice existant dans la console VPC Lattice. Pour plus d'informations, consultez la section Règles du récepteur pour votre service VPC Lattice.

11. (Facultatif) Pour configurer un équilibreur de charge pour votre service, développez Load balancing (Répartition de charge).

    Choisissez l'équilibreur de charge.

Pour utiliser cet équilibreur de charge	Faites ceci
Application Load Balancer	a. Pour Load balancer type (Type d'équilibreur de charge), sélection nez Application Load Balancer. b. Choisissez Create a new load balancer (Créer un équilibreur de charge) pour créer un équilibre ur de charge Application

Pour utiliser cet équilibreur de charge	Faites ceci	
	Load Balancer ou Use an existing load balancer (Utiliser un équilibreur de charge existant) pour sélectionner un équilibre ur de charge Application Load Balancer existant. c. Pour Load balancer name (nom d'équilibreur de charge), saisissez un nom unique. d. Pour Choose container to load balance (Choisissez le conteneur pour équilibre r la charge), choisissez le conteneur qui héberge le service. e. Pour Listener (Écouteur), saisissez un port et un protocole sur lesquels Application Load Balancer doit écouter les demandes de connexion. Par défaut, l'équilibreur de charge est configuré pour utiliser le port 80 et HTTP. f. Pour Target group name (Nom du groupe cible), saisissez un nom et un protocole pour le groupe cible vers lequel Application Load Balancer achemine les requêtes.	

Pour utiliser cet équilibreur de charge	Faites ceci	
	Par défaut, le groupe cible achemine les requêtes vers le premier conteneur défini dans votre définition de tâche. g. Pour le délai de désenregistrement, entrez le nombre de secondes pendant lequel l'équilibreur de charge doit passer à l'état cible. `UNUSED` La durée par défaut est 300 secondes. h. Pour Health check path (Chemin de surveillance de l'état), saisissez un chemin qui existe dans votre conteneur où Application Load Balancer doit envoyer périodiquement des requêtes pour vérifier l'état de la connexion entre Application Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (`/`).	

Pour utiliser cet équilibreur de charge	Faites ceci	
Network Load Balancer	a. Pour Load balancer type (Type d'équilibreur de charge), sélectionnez Network Load Balancer. b. Dans le champ Load Balancer (Équilibreur de charge), choisissez Attacher à un Network Load Balancer existant. c. Pour Choose container to load balance (Choisissez le conteneur pour équilibre r la charge), choisissez le conteneur qui héberge le service. d. Pour Target group name (Nom du groupe cible), saisissez un nom et un protocole pour le groupe cible vers lequel Network Load Balancer achemine les requêtes. Par défaut, le groupe cible achemine les requêtes vers le premier conteneur défini dans votre définition de tâche. e. Pour le délai de désenregi strement, entrez le nombre de secondes pendant lequel l'équilib reur de charge doit passer à l'état cible. `UNUSED`	

Pour utiliser cet équilibreur de charge	Faites ceci	
	La durée par défaut est 300 secondes. f. Pour Health check path (Chemin de surveillance de l'état), saisissez un chemin qui existe dans votre conteneur où Network Load Balancer doit envoyer périodiquement des requêtes pour vérifier l'état de la connexion entre Application Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (/).	

12. (Facultatif) Pour configurer le service Auto Scaling, développez Service auto scaling, puis spécifiez les paramètres suivants. Pour utiliser Predicte Auto Scaling, qui examine les données de charge passées issues des flux de trafic, configurez-le après avoir créé le service. Pour de plus amples informations, veuillez consulter Utilisez des modèles historiques pour faire évoluer les services Amazon ECS grâce à une mise à l'échelle prédictive.

    a. Pour utiliser la mise à l'échelle automatique du service, sélectionnez Service Auto Scaling (mise à l'échelle automatique du service).

    b. Dans le champ Nombre minimum de tâches, entrez la limite inférieure du nombre de tâches à utiliser pour le dimensionnement automatique du service. Le nombre souhaité ne sera pas inférieur à ce nombre.

    c. Dans le champ Nombre maximum de tâches, entrez la limite supérieure du nombre de tâches à utiliser pour le dimensionnement automatique du service. Le nombre souhaité ne sera pas supérieur à ce nombre.

    d. Choisissez le type de stratégie. Sous Type de politique de dimensionnement, choisissez l'une des options suivantes.

Pour utiliser ce type de politique	Faites ceci	
Suivi de la cible	a. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Target tracking (Suivi de cible). b. Pour Policy name (Nom de la politique), saisissez un nom de politique. c. Pour Métrique de service ECS, sélectionnez l'une des métriques suivantes. • ECSServiceMoyenneCPUUtilization : utilisation moyenne du processeur par le service. • ECSServiceAverageMemoryUtilization— Utilisation moyenne de la mémoire du service. • ALBRequestCountPerTarget— Nombre de demandes traitées par cible dans un groupe cible d'Application Load Balancer. d. PourTarget value (Valeur cible), entrez la valeur conservée par le service pour la métrique sélectionnée.	

Pour utiliser ce type de politique	Faites ceci	
	e. Pour la période de recharge, entrez le temps, en secondes, après une activité de scale-out (ajout de tâches) qui doit s'écouler avant qu'une autre activité de scale-out puisse démarrer. f. Pour la période de recharge progressive, entrez le temps, en secondes, après une activité de mise à l'échelle (suppression de tâches) qui doit s'écouler avant qu'une autre activité de mise à l'échelle ne puisse démarrer. g. Pour empêcher la politique d'effectuer une activité de mise à l'échelle horizontale, sélectionnez Turn off scale-in (Désactiver la mise à l'échelle horizontale). h. • (Facultatif) Sélectionnez Désactiver la mise à l'échelle si vous souhaitez que votre politique de dimension	

Pour utiliser ce type de politique	Faites ceci	
	nement soit adaptée à l'augmentation du trafic, mais que vous n'avez pas besoin qu'elle s'adapte lorsque le trafic diminue.	

Pour utiliser ce type de politique	Faites ceci	
Mise à l'échelle par étapes	a. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Mise à l'échelle par étapes. b. Pour Nom de la stratégie , saisissez un nom de stratégie. c. Pour Alarm name (Nom de l'alarme), entrez un nom unique pour l'alarme. d. Pour Métrique de service Amazon ECS, sélection nez la métrique à utiliser pour l'alarme. e. Pour Statistique, choisissez la statistique d'alarme. f. Pour Période, choisissez la période de l'alarme. g. Pour Condition d'alarme, choisissez comment comparer la métrique sélectionnée au seuil défini. h. Pour Seuil de comparais on des métriques et Période d'évaluat ion pour déclencher l'alarme, saisissez le seuil utilisé pour l'alarme	

Pour utiliser ce type de politique	Faites ceci	
	et la durée d'évaluation du seuil. i. Sous Actions de mise à l'échelle, procédez comme suit : • Pour Action, indiquez si vous souhaitez ajouter, supprimer ou définir un nombre spécifique souhaité pour votre service. • Si vous avez choisi d'ajouter ou de supprimer des tâches, dans Valeur, entrez le nombre de tâches (ou le pourcentage de tâches existante s) à ajouter ou à supprimer lorsque l'action de dimension nement est lancée. Si vous avez choisi de spécifier le nombre souhaité, saisissez le nombre de tâches. Pour Type, indiquez si la Valeur est un entier ou un pourcentage du nombre souhaité existant. • Pour Limite inférieur e et Limite supérieur	

Pour utiliser ce type de politique	Faites ceci	
	e, saisissez les limites inférieure et supérieur e de votre ajustement de mise à l'échelle par étapes. Par défaut, la limite inférieure pour l'ajout d'une politique est le seuil de l'alarme et la limite supérieur e est l'infini positif (+). Par défaut, la limite supérieure pour la suppression d'une politique est le seuil de l'alarme et la limite inférieure est l'infini négatif (-). • (Facultatif) Ajoutez des options de mise à l'échelle supplémen taires. Choisissez Ajouter une nouvelle action de dimension nement, puis répétez les étapes des actions de dimensionnement. • Pour la période de recharge, entrez le temps, en secondes, nécessaire pour qu'une activité de dimensionnement précédente prenne	

Pour utiliser ce type de politique	Faites ceci	
	effet. Dans le cas d'une politique d'ajout, il s'agit du moment où, après une activité de scale-out, la politique de scale-out bloque les activités de scale-in et limite le nombre de tâches pouvant être redimensionnées à la fois. Pour une politique de suppression, il s'agit de la période qui suit une activité d'extension qui doit se terminer avant qu'une autre activité d'extension ne puisse démarrer.	

13. (Facultatif) Pour utiliser une stratégie de placement des tâches autre que la stratégie par défaut, développez Task Placement (Placement des tâches), puis choisissez parmi les options suivantes.

    Pour de plus amples informations, veuillez consulter Comment Amazon ECS place les tâches sur les instances de conteneur.

    - Répartition équilibrée AZ : répartissez les tâches entre les zones de disponibilité et entre les instances de conteneur de la zone de disponibilité.
    - AZ Balanced BinPack — Répartissez les tâches entre les zones de disponibilité et entre les instances de conteneur disposant du moins de mémoire disponible.
    - BinPack— Répartissez les tâches en fonction de la quantité minimale de processeur ou de mémoire disponible.

- Une tâche par hôte : placez au maximum une tâche du service sur chaque instance de conteneur.
- Personnalisé : définissez votre propre stratégie de placement des tâches.

Si vous avez choisi Custom (Personnaliser), définissez l'algorithme de placement des tâches et les règles à prendre en compte lors du placement des tâches.

- Sous Strategy (Stratégie), pour Type et Field (Champ), choisissez l'algorithme et l'entité à utiliser pour l'algorithme.

  Vous pouvez saisir jusqu'à 5 stratégies maximum.
- Sous Contrainte, pour Type et Expression, choisissez la règle et l'attribut pour la contrainte.

  Par exemple, pour définir la contrainte permettant de placer des tâches sur des instances T2, pour Expression, saisissez attribute:ecs.instance-type =~ t2.*.

  Vous pouvez saisir jusqu'à 10 contraintes maximum.

14. Si votre tâche utilise un volume de données compatible avec la configuration lors du déploiement, vous pouvez configurer le volume en développant Volume.

    Le nom et le type de volume sont configurés lorsque vous créez une révision de définition de tâche et ne peuvent pas être modifiés lors de la création d'un service. Pour mettre à jour le nom et le type du volume, vous devez créer une nouvelle révision de définition de tâche et créer un service à l'aide de cette nouvelle révision.

Pour configurer ce type de volume	Faites ceci	
Amazon EBS	a. Pour le type de volume EBS, choisissez le type de volume EBS que vous souhaitez associer à votre tâche. b. Pour Taille (GiB), entrez une valeur valide pour la taille du volume en gibioctets (GiB). Vous	

Pour configurer ce type de volume	Faites ceci	
	pouvez spécifier une taille de volume minimale de 1 GiB et maximale de 16 384 GiB. Cette valeur est obligatoire sauf si vous fournissez un identifiant de capture d'écran. c. Pour les IOPS, entrez le nombre maximum d' input/output opérations (IOPS) que le volume doit fournir. Cette valeur est configurable uniquemen t pour les types de `gp3` volume `io1io2`, et. d. Pour Débit (Mib/s), entrez le débit que le volume doit fournir, en mégaoctets par seconde (ou Mib/s). MiBps Cette valeur est configura ble uniquement pour le type de `gp3` volume. e. Pour Snapshot ID, choisissez un instantané de volume Amazon EBS existant ou entrez l'ARN d'un instantané si vous souhaitez créer un volume à partir d'un instantané. Vous pouvez également créer un nouveau volume vide en ne choisissant ni	

Pour configurer ce type de volume	Faites ceci	
	en saisissant un identifiant de capture d'écran. f. Si vous spécifiez un ID de capture d'écran, vous pouvez spécifier un taux d'initialisation du volume (Mbits/s). Entrez une valeur comprise entre 100 et 300, en Mbits/s, qui déterminera la vitesse de chargement des données à partir de l'instantané spécifié à l'aide de l'ID de capture pour la création du volume. g. Pour Type de système de fichiers, choisissez le type de système de fichiers qui sera utilisé pour le stockage et la récupération des données sur le volume. Vous pouvez choisir le système d'exploitation par défaut ou un type de système de fichiers spécifique. La valeur par défaut pour Linux est`XFS`. Pour les volumes créés à partir d'un instantané, vous devez spécifier le même type de système de fichiers que celui	

Pour configurer ce type de volume	Faites ceci	
	utilisé par le volume lors de la création de l'instantané. Si le type de système de fichiers ne correspond pas, la tâche ne démarrera pas. h. Pour le rôle d'infrastructure, choisissez un rôle IAM doté des autorisations nécessaires pour permettre à Amazon ECS de gérer les volumes Amazon EBS pour les tâches. Vous pouvez associer la politique `AmazonECSInfrastructureRolePolicyForVolumes` gérée au rôle, ou vous pouvez utiliser la politique comme guide pour créer et associer votre propre politique avec des autorisations répondant à vos besoins spécifiques. Pour plus d'informations sur les autorisations nécessaires, consultez Rôle IAM dans l'infrastructure Amazon ECS. i. Pour le chiffrement, choisissez Par défaut si vous souhaitez utiliser	

Pour configurer ce type de volume	Faites ceci	
	le chiffrement Amazon EBS par défaut. Si le chiffrement est configuré par défaut sur votre compte, le volume sera chiffré avec la clé AWS Key Management Service (AWS KMS) spécifiée dans le paramètre. Si vous choisissez Par défaut et que le chiffrement par défaut d'Amazon EBS n'est pas activé, le volume ne sera pas chiffré. Si vous choisissez Personnalisé, vous pouvez spécifier celui AWS KMS key de votre choix pour le chiffrement du volume. Si vous choisissez Aucun, le volume ne sera pas chiffré, sauf si le chiffrement est configuré par défaut ou si vous créez un volume à partir d'un instantané chiffré. j. Si vous avez choisi Personnalisé pour le chiffrement, vous devez spécifier celui AWS KMS	

Pour configurer ce type de volume	Faites ceci	
	key que vous souhaitez utiliser. Pour la clé KMS, choisissez AWS KMS key ou entrez un ARN de clé. Si vous choisissez de chiffrer votre volume à l'aide d'une clé symétriqu e gérée par le client, assurez-vous que vous disposez des autorisat ions appropriées définies dans votre AWS KMS key politique. Pour plus d'informations, consultez la section Chiffrement des données pour les volumes Amazon EBS. k. (Facultatif) Sous Balises, vous pouvez ajouter des balises à votre volume Amazon EBS soit en propageant des balises à partir de la définition de la tâche ou du service, soit en fournissant vos propres balises. Si vous souhaitez propager des balises à partir de la définition de tâche, choisissez Définitio n de tâche pour propager des balises à partir de. Si vous souhaitez propager	

Pour configurer ce type de volume	Faites ceci	
	des balises depuis le service, choisissez Service for Propagate tags from. Si vous choisisse z Ne pas propager, ou si vous ne choisissez aucune valeur, les balises ne sont pas propagées. Si vous souhaitez fournir vos propres balises, choisissez Ajouter une balise, puis indiquez la clé et la valeur pour chaque balise que vous ajoutez. Pour plus d'informa tions sur le balisage des volumes Amazon EBS, consultez la section Marquage des volumes Amazon EBS.	

15. (Facultatif) Pour vous aider à identifier votre service et vos tâches, développez Tags (balises), puis configurez vos balises.

    Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises de définition des tâches, sélectionnez Activer les balises gérées Amazon ECS, puis pour Propager des balises à partir de, choisissez Définitions de tâches.

    Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises d'un service, sélectionnez Activer les balises gérées Amazon ECS, puis pour Propager des balises à partir de, choisissez Service.

    Ajoutez ou supprimez une balise.

- [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :
  - Pour Clé, saisissez le nom de la clé.
  - Pour Valeur, saisissez la valeur de clé.
- [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.

16. Choisissez Créer.

Étapes suivantes

Vous trouverez ci-dessous des actions supplémentaires après avoir créé un service.

- Configurez Predicte Auto Scaling, qui examine les données de charge passées issues des flux de trafic. Pour de plus amples informations, veuillez consulter Utilisez des modèles historiques pour faire évoluer les services Amazon ECS grâce à une mise à l'échelle prédictive.
- Suivez votre déploiement et consultez l'historique de vos services pour les services qu'Amazon ECS disjoncte. Pour de plus amples informations, veuillez consulter Afficher l'historique des services à l'aide des déploiements de services Amazon ECS.

blue/green Déploiements Amazon ECS

Un blue/green déploiement est une méthodologie de publication qui réduit les temps d'arrêt et les risques en exécutant deux environnements de production identiques appelés bleu et vert. Avec les blue/green déploiements Amazon ECS, vous pouvez valider les nouvelles révisions des services avant de diriger le trafic de production vers celles-ci. Cette approche fournit un moyen plus sûr de déployer les modifications avec la possibilité de revenir rapidement en arrière si nécessaire.

Avantages

Les avantages liés à l'utilisation des blue/green déploiements sont les suivants :

- Réduit les risques en testant le trafic de production avant de passer à la production. Vous pouvez valider le nouveau déploiement avec le trafic de test avant de diriger le trafic de production vers celui-ci.
- Déploiements sans interruption de service. L'environnement de production reste disponible tout au long du processus de déploiement, garantissant ainsi la disponibilité continue du service.

- Annulation facile si des problèmes sont détectés. Si des problèmes surviennent avec le déploiement vert, vous pouvez rapidement revenir au déploiement bleu sans interruption de service prolongée.
- Environnement de test contrôlé. L'environnement vert fournit un espace isolé pour tester les nouvelles fonctionnalités avec des modèles de trafic réels avant le déploiement complet.
- Processus de déploiement prévisible. L'approche structurée avec des étapes du cycle de vie définies rend les déploiements plus cohérents et fiables.
- Validation automatisée via des hooks du cycle de vie. Vous pouvez implémenter des tests automatisés à différentes étapes du déploiement afin de vérifier le fonctionnement.

Terminologie

Les conditions de blue/green déploiement d'Amazon ECS sont les suivantes :

- Temps de cuisson : durée pendant laquelle les révisions du service bleu et vert sont exécutées simultanément après le déplacement du trafic de production.
- Déploiement bleu : révision actuelle du service de production que vous souhaitez remplacer.
- Déploiement écologique : nouvelle version du service que vous souhaitez déployer.
- Étapes du cycle de vie : série d'événements survenant au cours de l'opération de déploiement, tels que « après le transfert du trafic de production ».
- Lifecycle Hook : fonction Lambda qui vérifie le déploiement à une étape spécifique du cycle de vie.
- Listener : ressource Elastic Load Balancing qui vérifie les demandes de connexion à l'aide du protocole et du port que vous configurez. Les règles que vous définissez pour un écouteur déterminent la manière dont Amazon ECS achemine les demandes vers ses cibles enregistrées.
- Règle : ressource Elastic Load Balancing associée à un écouteur. Une règle définit la manière dont les demandes sont acheminées et comprend une action, une condition et une priorité.
- Groupe cible : ressource Elastic Load Balancing utilisée pour acheminer les demandes vers une ou plusieurs cibles enregistrées (par exemple, EC2 des instances). Lorsque vous créez un écouteur, vous spécifiez un groupe cible pour son action par défaut. Le trafic est transféré vers le groupe cible spécifié dans la règle de l'écouteur.
- Transfert de trafic : processus utilisé par Amazon ECS pour transférer le trafic du déploiement bleu au déploiement vert. Pour les blue/green déploiements Amazon ECS, tout le trafic passe du service bleu au service vert en une seule fois.

Considérations

Tenez compte des points suivants lorsque vous choisissez un type de déploiement :

- Utilisation des ressources : Blue/green les déploiements exécutent temporairement les révisions des services bleu et vert simultanément, ce qui peut doubler votre utilisation des ressources pendant les déploiements.
- Surveillance du déploiement : Blue/green les déploiements fournissent des informations plus détaillées sur l'état du déploiement, ce qui vous permet de surveiller chaque étape du processus de déploiement.
- Annulation : Blue/green les déploiements facilitent le retour à la version précédente en cas de détection de problèmes, car la version bleue est maintenue en cours d'exécution jusqu'à l'expiration du délai de cuisson.

Flux de travail des déploiements blue/green de services Amazon ECS

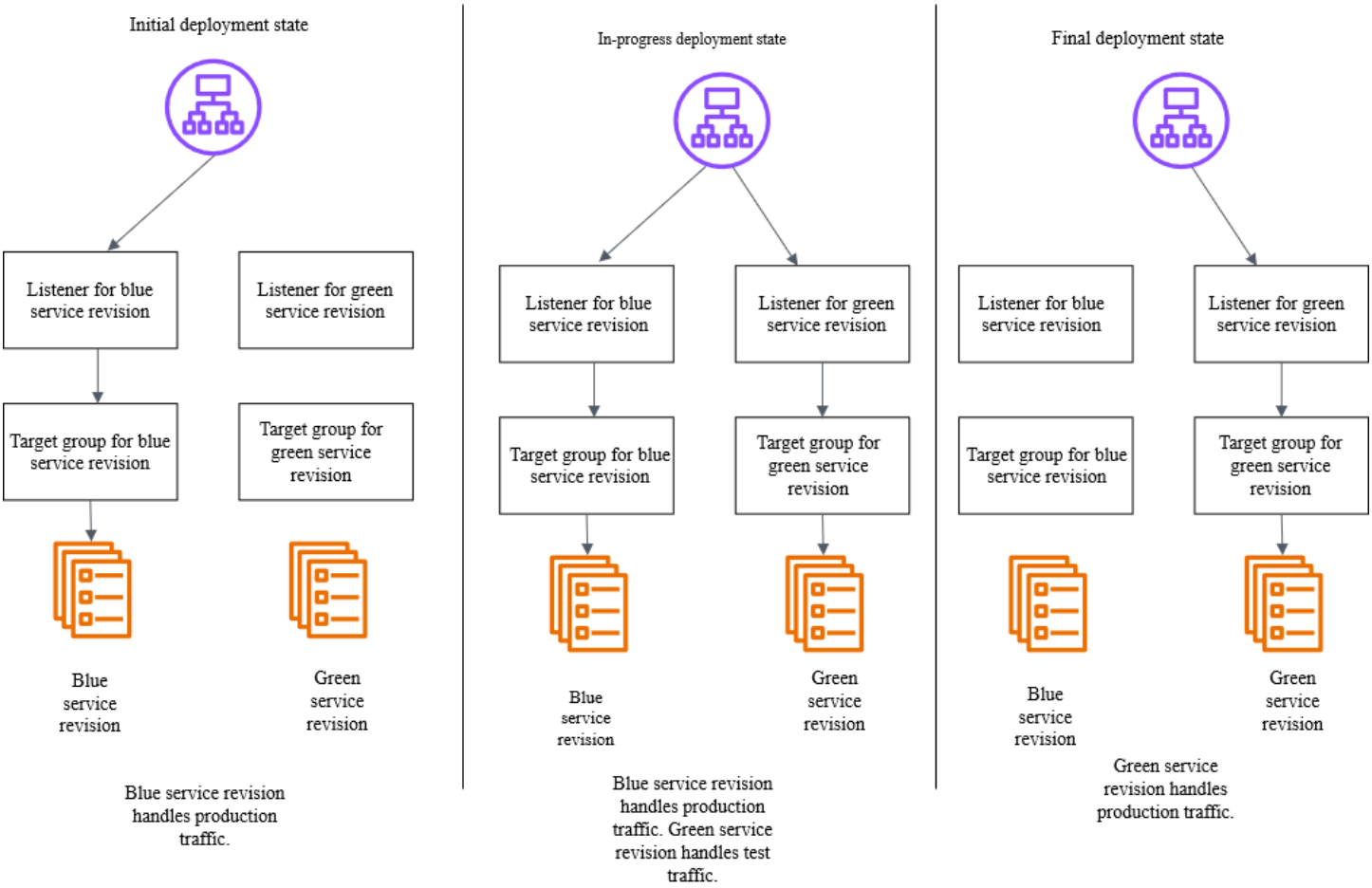
Le processus de blue/green déploiement d'Amazon ECS suit une approche structurée comportant six phases distinctes qui garantissent des mises à jour d'applications sûres et fiables. Chaque phase a un objectif spécifique en validant et en faisant passer votre application de la version actuelle (bleu) à la nouvelle version (vert).

1. Phase de préparation : créer l'environnement vert à côté de l'environnement bleu existant. Cela inclut la mise à disposition de nouvelles révisions de service et la préparation des groupes cibles.
2. Phase de déploiement : Déployez la nouvelle version du service dans un environnement écologique. Amazon ECS lance de nouvelles tâches à l'aide de la révision du service mise à jour tandis que l'environnement bleu continue de desservir le trafic de production.
3. Phase de test : valider l'environnement écologique à l'aide du routage du trafic de test. L'Application Load Balancer dirige les demandes de test vers l'environnement vert tandis que le trafic de production reste en mode bleu.
4. Phase de transfert du trafic : déplacez le trafic de production du bleu au vert en fonction de votre stratégie de déploiement configurée. Cette phase inclut des points de contrôle de surveillance et de validation.
5. Phase de surveillance : surveillez l'état de l'application, les indicateurs de performance et les états d'alarme pendant la période de cuisson. Une opération de restauration est lancée lorsque des problèmes sont détectés.

6. Phase d'achèvement : finalisez le déploiement en mettant fin à l'environnement bleu ou en le maintenant pour d'éventuels scénarios de restauration, en fonction de votre configuration.

Flux de travail

Le schéma suivant illustre le flux de travail de blue/green déploiement complet, illustrant l'interaction entre Amazon ECS et l'Application Load Balancer :



Le flux de travail de déploiement amélioré inclut les étapes détaillées suivantes :

1. État initial : Le service bleu (production actuelle) gère 100 % du trafic de production. L'Application Load Balancer possède un écouteur unique avec des règles qui acheminent toutes les demandes vers le groupe cible bleu contenant des tâches bleues saines.
2. Provisionnement d'un environnement écologique : Amazon ECS crée de nouvelles tâches à l'aide de la définition de tâche mise à jour. Ces tâches sont enregistrées auprès d'un nouveau groupe cible vert mais ne reçoivent aucun trafic au départ.

3. Validation du bilan de santé : l'Application Load Balancer effectue des contrôles de santé sur les tâches vertes. Ce n'est que lorsque les tâches écologiques passent aux tests de santé que le déploiement passe à la phase suivante.
4. Routage du trafic de test : si elles sont configurées, les règles d'écoute de l'équilibreur de charge d'application acheminent des modèles de trafic spécifiques (tels que les demandes avec des en-têtes de test) vers l'environnement vert pour validation tandis que le trafic de production reste en bleu. Ceci est contrôlé par le même écouteur qui gère le trafic de production, en utilisant des règles différentes basées sur les attributs des demandes.
5. Transfert du trafic de production : en fonction de la configuration du déploiement, le trafic passe du bleu au vert. Dans les blue/green déploiements ECS, il s'agit d'un changement immédiat (all-at-once) où 100 % du trafic est transféré de l'environnement bleu vers l'environnement vert. L'Application Load Balancer utilise un écouteur unique avec des règles d'écoute qui contrôlent la distribution du trafic entre les groupes cibles bleus et verts en fonction des pondérations.
6. Surveillance et validation : tout au long du transfert de trafic, Amazon ECS surveille les CloudWatch métriques, les états d'alarme et l'état du déploiement. Les déclencheurs de restauration automatique s'activent si des problèmes sont détectés.
7. Période de cuisson : durée pendant laquelle les révisions du service bleu et vert sont exécutées simultanément après le déplacement du trafic de production.
8. Fin de l'environnement bleu : une fois le transfert de trafic et la validation réussis, l'environnement bleu est arrêté pour libérer les ressources du cluster, ou maintenu pour permettre une restauration rapide.
9. État final : L'environnement vert devient le nouvel environnement de production, gérant 100 % du trafic. Le déploiement est marqué comme réussi.

Étapes du cycle de vie du

Le processus de blue/green déploiement passe par des étapes distinctes du cycle de vie (une série d'événements au cours de l'opération de déploiement, tels que « après le transfert du trafic de production »), chacune comportant des responsabilités et des points de contrôle de validation spécifiques. La compréhension de ces étapes vous permet de suivre la progression du déploiement et de résoudre les problèmes de manière efficace.

Chaque étape du cycle de vie peut durer jusqu'à 24 heures. Nous recommandons que la valeur reste inférieure à 24 heures. Cela est dû au fait que les processus asynchrones ont besoin de temps pour déclencher les hooks. Le système expire, échoue au déploiement, puis lance une restauration après une période de 24 heures. AWS CloudFormation les déploiements sont soumis à des restrictions

de délai d'expiration supplémentaires. Bien que la limite de 24 heures reste en vigueur, elle AWS CloudFormation impose une limite de 36 heures à l'ensemble du déploiement. AWS CloudFormation échoue au déploiement, puis lance une annulation si le processus ne se termine pas dans les 36 heures.

Étapes du cycle de	Description	Utiliser cette étape pour Lifecycle Hook ?
SERVICE DE RÉCONCILIATION	Cette étape se produit uniquement lorsque vous démarrez un nouveau déploiement de service avec plus d'une révision de service dans un état ACTIF.	Oui
PRÉ_SCALE_UP	La révision du service écologique n'a pas encore commencé. La révision du service bleu gère 100 % du trafic de production. Il n'y a aucun trafic de test.	Oui
SCALE_UP	Le moment où la révision du service écologique passe à 100 % et lance de nouvelles tâches. La révision du service vert ne dessert aucun trafic pour le moment.	Non
POST_SCALE_UP	La révision du service écologique a commencé. La révision du service bleu gère 100 % du trafic de production. Il n'y a aucun trafic de test.	Oui
TEST_TRAFFIC_SHIFT	Les révisions des services bleu et vert sont en cours. La révision du service bleu gère 100 % du trafic de production. La révision du service vert passe de 0 % à 100 % du trafic de test.	Oui

Étapes du cycle de	Description	Utiliser cette étape pour Lifecycle Hook ?
POST-TEST_TRAFFIC_SHIFT	Le changement de trafic d'essai est terminé. La révision du service écologique gère 100 % du trafic de test.	Oui
PRODUCTION_TRAFFIC _SHIFT	Le trafic de production est en train de passer à la révision des services écologiques. La révision du service écologique passe de 0 % à 100 % du trafic de production.	Oui
POST-PRODUCTION TRAFFIC_SHIFT	Le transfert du trafic de production est terminé.	Oui
HEURE DE CUISSON	Durée pendant laquelle les révisions du service bleu et vert sont exécutées simultanément.	Non
NETTOYAGE	La révision du service bleu a été complètement réduite à 0 tâches en cours d'exécution. La révision du service écologique est désormais la révision du service de production après cette étape.	Non

Chaque étape du cycle de vie inclut des points de contrôle de validation intégrés qui doivent passer avant de passer à l'étape suivante. Si une validation échoue, le déploiement peut être automatiquement annulé pour garantir la disponibilité et la fiabilité du service.

Ressources requises pour les blue/green déploiements Amazon ECS

Pour utiliser un blue/green déploiement avec transfert de trafic géré, votre service doit utiliser l'une des fonctionnalités suivantes :

- Elastic Load Balancing
- Service Connect

Les services qui n'utilisent pas Service Discovery, Service Connect, VPC Lattice ou Elastic Load Balancing peuvent également utiliser des blue/green déploiements, mais ne bénéficient d'aucun des avantages de la gestion du transfert de trafic.

La liste suivante fournit un aperçu général de ce que vous devez configurer pour les blue/green déploiements Amazon ECS :

- Votre service utilise un Application Load Balancer, un Network Load Balancer ou un Service Connect. Configurez les ressources appropriées.
  - Application Load Balancer - Pour plus d'informations, consultez. Ressources d'Application Load Balancer pour les déploiements blue/green
  - Network Load Balancer - Pour plus d'informations, consultez. Ressources Network Load Balancer pour les déploiements bleu/vert d'Amazon ECS
  - Service Connect - Pour plus d'informations, consultezRessources Service Connect pour les déploiements bleu/vert d'Amazon ECS.
- Définissez le contrôleur de déploiement des services sur`ECS`.
- Configurez la stratégie de déploiement `blue/green` conformément à votre définition de service.
- Configurez éventuellement des paramètres supplémentaires tels que :
  - Il est temps de préparer le nouveau déploiement
  - CloudWatch alarmes pour annulation automatique
  - Crochets du cycle de vie du déploiement pour les tests (il s'agit de fonctions Lambda qui s'exécutent à des étapes de déploiement spécifiées)

Bonnes pratiques

Suivez ces bonnes pratiques pour réussir vos blue/green déploiements Amazon ECS :

- Configurez les contrôles de santé appropriés qui reflètent précisément l'état de santé de votre application.
- Définissez un temps de cuisson permettant de tester suffisamment le déploiement écologique.
- Implémentez des CloudWatch alarmes pour détecter automatiquement les problèmes et déclencher des annulations.
- Utilisez les hooks du cycle de vie pour effectuer des tests automatisés à chaque étape du déploiement.

- Assurez-vous que votre application peut gérer les environnements bleu et vert exécutés simultanément.
- Prévoyez une capacité de cluster suffisante pour gérer les deux environnements lors du déploiement.
- Testez vos procédures de rollback avant de les mettre en œuvre en production.

Ressources d'Application Load Balancer pour les déploiements blue/green

Pour utiliser les équilibreurs de charge d'application avec les blue/green déploiements Amazon ECS, vous devez configurer des ressources spécifiques qui permettent le routage du trafic entre les révisions de service bleues et vertes.

Limites

Les limitations suivantes s'appliquent actuellement aux blue/green déploiements avec des équilibreurs de charge d'application :

- Les déploiements bleu/vert ne sont pas pris en charge lors de l'utilisation des écouteurs Application Load Balancer dont plusieurs types d'actions sont configurés avec des actions autres que le transfert. Si les règles de votre écouteur Application Load Balancer contiennent une combinaison d'actions de transfert et de non-transfert (telles que des actions de redirection, de réponse fixe ou d'authentification), le déploiement échouera.

Groupes cibles

Pour les blue/green déploiements avec Elastic Load Balancing, vous devez créer deux groupes cibles :

- Un groupe cible principal pour la révision du service bleu (trafic de production actuel)
- Un autre groupe cible pour la révision du service écologique (nouvelle version)

Les deux groupes cibles doivent être configurés avec les paramètres suivants :

- Type de cible : `IP` (pour Fargate EC2 ou `awsvpc` en mode réseau)
- Protocole : `HTTP` (ou le protocole utilisé par votre application)
- Port : port sur lequel votre application écoute (généralement `80` pour le protocole HTTP)
- VPC : le même VPC que vos tâches Amazon ECS

- Paramètres de vérification de l'état : configurés pour vérifier correctement l'état de santé de votre application

Au cours d'un blue/green déploiement, Amazon ECS enregistre automatiquement les tâches auprès du groupe cible approprié en fonction de l'étape de déploiement.

Example Création de groupes cibles pour un Application Load Balancer

Les commandes CLI suivantes créent deux groupes cibles à utiliser avec un Application Load Balancer dans le cadre d'un blue/green déploiement :

```
aws elbv2 create-target-group \
    --name blue-target-group \
    --protocol HTTP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-path / \
    --health-check-protocol HTTP \
    --health-check-interval-seconds 30 \
    --health-check-timeout-seconds 5 \
    --healthy-threshold-count 2 \
    --unhealthy-threshold-count 2

aws elbv2 create-target-group \
    --name green-target-group \
    --protocol HTTP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-path / \
    --health-check-protocol HTTP \
    --health-check-interval-seconds 30 \
    --health-check-timeout-seconds 5 \
    --healthy-threshold-count 2 \
    --unhealthy-threshold-count 2
```

Application Load Balancer

Vous devez créer un Application Load Balancer avec la configuration suivante :

- Schéma : connecté à Internet ou interne, selon vos besoins

- Type d'adresse IP : IPv4
- VPC : le même VPC que vos tâches Amazon ECS
- Sous-réseaux : au moins deux sous-réseaux dans des zones de disponibilité différentes
- Groupes de sécurité : groupe de sécurité qui autorise le trafic sur les ports de l'écouteur

Le groupe de sécurité attaché à l'Application Load Balancer doit disposer d'une règle sortante qui autorise le trafic vers le groupe de sécurité associé à vos tâches Amazon ECS.

Example Création d'un équilibreur de charge Application Load Balancer

La commande CLI suivante crée un Application Load Balancer à utiliser dans un déploiement bleu/vert :

```
aws elbv2 create-load-balancer \
    --name my-application-load-balancer \
    --type application \
    --security-groups sg-abcd1234 \
    --subnets subnet-12345678 subnet-87654321
```

Écouteurs et règles

Pour les blue/green déploiements, vous devez configurer des écouteurs sur votre Application Load Balancer :

- Écouteur de production : gère le trafic de production (généralement sur le port 80 ou 443)
  - Transfère initialement le trafic au groupe cible principal (révision du service bleu)
  - Après le déploiement, transfert le trafic vers le groupe cible alternatif (révision du service écologique)
- Écouteur de test (facultatif) : gère le trafic de test pour valider la révision du service écologique avant de transférer le trafic de production
  - Peut être configuré sur un port différent (par exemple, 8080 ou 8443)
  - Transfère le trafic vers le groupe cible alternatif (révision du service écologique) pendant les tests

Au cours d'un blue/green déploiement, Amazon ECS met automatiquement à jour les règles du récepteur pour acheminer le trafic vers le groupe cible approprié en fonction de la phase de déploiement.

Example Création d'un écouteur de production

La commande CLI suivante crée un écouteur de production sur le port 80 qui transmet le trafic au groupe cible principal (bleu) :

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
app/my-application-load-balancer/abcdef123456 \
    --protocol HTTP \
    --port 80 \
    --default-actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
blue-target-group/abcdef123456
```

Example Création d'un écouteur de test

La commande CLI suivante crée un écouteur de test sur le port 8080 qui transmet le trafic au groupe cible alternatif (vert) :

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
app/my-application-load-balancer/abcdef123456 \
    --protocol HTTP \
    --port 8080 \
    --default-actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
green-target-group/ghijkl789012
```

Example Création d'une règle d'écoute pour le routage basé sur le chemin

La commande CLI suivante crée une règle qui transmet le trafic d'un chemin spécifique au groupe cible vert à des fins de test :

```
aws elbv2 create-rule \
    --listener-arn arn:aws:elasticloadbalancing:region:123456789012:listener/app/my-
application-load-balancer/abcdef123456/ghijkl789012 \
    --priority 10 \
    --conditions Field=path-pattern,Values='/test/*' \
    --actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
green-target-group/ghijkl789012
```

Example Création d'une règle d'écoute pour le routage basé sur les en-têtes

La commande CLI suivante crée une règle qui transmet le trafic avec un en-tête spécifique au groupe cible vert à des fins de test :

```
aws elbv2 create-rule \
    --listener-arn arn:aws:elasticloadbalancing:region:123456789012:listener/app/my-application-load-balancer/abcdef123456/ghijkl789012 \
    --priority 20 \
    --conditions Field=http-header,HttpHeaderConfig='{Name=X-Environment,Values=[test]}' \
    --actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/green-target-group/ghijkl789012
```

Configuration du service

Vous devez disposer des autorisations nécessaires pour autoriser Amazon ECS à gérer les ressources de l'équilibreur de charge dans vos clusters en votre nom. Pour de plus amples informations, veuillez consulter Rôle IAM de l'infrastructure Amazon ECS pour les équilibreurs de charge.

Lorsque vous créez ou mettez à jour un service Amazon ECS pour blue/green des déploiements avec Elastic Load Balancing, vous devez spécifier la configuration suivante.

Remplacez le `user-input` par vos valeurs.

Les principaux composants de cette configuration sont les suivants :

- `targetGroupArn`: ARN du groupe cible principal (révision du service bleu).
- `alternateTargetGroupArn`: ARN du groupe cible alternatif (révision du service vert).
- `productionListenerRule`: l'ARN de la règle d'écoute pour le trafic de production.
- `roleArn`: ARN du rôle qui permet à Amazon ECS de gérer les ressources Elastic Load Balancing.
- `strategy`: défini sur pour activer `BLUE_GREEN` les déploiements bleu/vert.
- `bakeTimeInMinutes`: durée pendant laquelle les révisions du service bleu et vert sont exécutées simultanément après le déplacement du trafic de production.
- `TestListenerRule`: l'ARN de la règle d'écoute pour le trafic de test. Ce paramètre est facultatif.

```
{
```

```
    "loadBalancers": [
        {
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/primary-target-group/
abcdef123456",
            "containerName": "container-name",
            "containerPort": 80,
            "advancedConfiguration": {
                "alternateTargetGroupArn":
 "arn:aws:elasticloadbalancing:region:account-id:targetgroup/alternate-target-group/
ghijkl789012",
                "productionListenerRule": "arn:aws:elasticloadbalancing:region:account-
id:listener-rule/app/load-balancer-name/abcdef123456/listener/ghijkl789012/rule/
mnopqr345678",
                "roleArn": "arn:aws:iam::123456789012:role/ecs-elb-role"
            }
        }
    ],
    "deploymentConfiguration": {
        "strategy": "BLUE_GREEN",
        "maximumPercent": 200,
        "minimumHealthyPercent": 100,
        "bakeTimeInMinutes": 5
    }
}
```

Flux de trafic pendant le déploiement

Lors d'un blue/green déploiement avec Elastic Load Balancing, le trafic circule dans le système comme suit :

1. État initial : tout le trafic de production est acheminé vers le groupe cible principal (révision du service bleu).
2. Déploiement de la révision du service écologique : Amazon ECS déploie les nouvelles tâches et les enregistre auprès du groupe cible alternatif.
3. Trafic de test : si un écouteur de test est configuré, le trafic de test est acheminé vers le groupe cible alternatif pour valider la révision du service vert.
4. Transfert du trafic de production : Amazon ECS met à jour la règle du récepteur de production pour acheminer le trafic vers le groupe cible alternatif (révision du service vert).
5. Temps de cuisson : durée pendant laquelle les révisions du service bleu et vert sont exécutées simultanément après le déplacement du trafic de production.

6. Achèvement : Après un déploiement réussi, la révision du service bleu est interrompue.

Si des problèmes sont détectés lors du déploiement, Amazon ECS peut automatiquement revenir en arrière en redirigeant le trafic vers le groupe cible principal (révision du service bleu).

Ressources Network Load Balancer pour les déploiements bleu/vert d'Amazon ECS

Pour utiliser un Network Load Balancer avec les blue/green déploiements Amazon ECS, vous devez configurer des ressources spécifiques qui permettent le routage du trafic entre les révisions de service bleues et vertes. Cette section explique les composants requis et leur configuration.

Lorsque votre configuration inclut un Network Load Balancer, Amazon ECS ajoute un délai de 10 minutes aux étapes du cycle de vie suivantes :

- PRÉ_SCALE_UP
- TEST_TRAFFIC_SHIFT
- PRODUCTION_TRAFFIC_SHIFT

Ce délai est à l'origine des problèmes de synchronisation du Network Load Balancer, qui peuvent entraîner une inadéquation entre les pondérations de trafic configurées et le routage réel du trafic dans le plan de données.

Groupes cibles

Pour les blue/green déploiements avec un Network Load Balancer, vous devez créer deux groupes cibles :

- Un groupe cible principal pour la révision du service bleu (trafic de production actuel)
- Un autre groupe cible pour la révision du service écologique (nouvelle révision du service)

Les deux groupes cibles doivent être configurés avec les paramètres suivants :

- Type de cible : `ip` (pour Fargate EC2 ou `awsvpc` en mode réseau)
- Protocole : `TCP` (ou le protocole utilisé par votre application)
- Port : port sur lequel votre application écoute (généralement `80` pour le protocole HTTP)
- VPC : le même VPC que vos tâches Amazon ECS

- Paramètres de vérification de l'état : configurés pour vérifier correctement l'état de santé de votre application

  Pour les vérifications de l'état du protocole TCP, le Network Load Balancer établit une connexion TCP avec la cible. Si la connexion est réussie, la cible est considérée comme saine.

  Pour les bilans de HTTP/HTTPS santé, le Network Load Balancer envoie une HTTP/HTTPS demande à la cible et vérifie la réponse.

Au cours d'un blue/green déploiement, Amazon ECS enregistre automatiquement les tâches auprès du groupe cible approprié en fonction de l'étape de déploiement.

Example Création de groupes cibles pour un Network Load Balancer

Les commandes AWS CLI suivantes créent deux groupes cibles à utiliser avec un Network Load Balancer dans le cadre d'un blue/green déploiement :

```
aws elbv2 create-target-group \
    --name blue-target-group \
    --protocol TCP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-protocol TCP

aws elbv2 create-target-group \
    --name green-target-group \
    --protocol TCP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-protocol TCP
```

Network Load Balancer

Vous devez créer un Network Load Balancer avec la configuration suivante :

- Schéma : connecté à Internet ou interne, selon vos besoins
- Type d'adresse IP : IPv4
- VPC : le même VPC que vos tâches Amazon ECS

- Sous-réseaux : au moins deux sous-réseaux dans des zones de disponibilité différentes

Contrairement aux équilibreurs de charge d'application, les équilibreurs de charge réseau fonctionnent au niveau de la couche transport (couche 4) et n'utilisent pas de groupes de sécurité. Vous devez plutôt vous assurer que les groupes de sécurité associés à vos tâches Amazon ECS autorisent le trafic provenant du Network Load Balancer sur les ports du récepteur.

Example Créer un Network Load Balancer

La commande AWS CLI suivante crée un Network Load Balancer à utiliser dans le cadre d'un blue/green déploiement :

```
aws elbv2 create-load-balancer \
    --name my-network-load-balancer \
    --type network \
    --subnets subnet-12345678 subnet-87654321
```

Considérations relatives à l'utilisation du NLB dans le cadre de déploiements blue/green

Lorsque vous utilisez un Network Load Balancer pour les blue/green déploiements, tenez compte des points suivants :

- Fonctionnement de couche 4 : les équilibreurs de charge réseau fonctionnent au niveau de la couche transport (couche 4) et n'inspectent pas le contenu de la couche application (couche 7). Cela signifie que vous ne pouvez pas utiliser les en-têtes ou les chemins HTTP pour les décisions de routage.
- Contrôles de santé : les contrôles de santé de Network Load Balancer sont limités aux protocoles TCP, HTTP ou HTTPS. Pour les vérifications de l'état du protocole TCP, le Network Load Balancer vérifie uniquement que la connexion peut être établie.
- Préservation de la connexion : les équilibreurs de charge réseau préservent l'adresse IP source du client, ce qui peut être utile à des fins de sécurité et de journalisation.
- Adresses IP statiques : les équilibreurs de charge réseau fournissent des adresses IP statiques pour chaque sous-réseau, ce qui peut être utile pour les listes blanches ou lorsque les clients doivent se connecter à une adresse IP fixe.
- Trafic de test : étant donné que les équilibreurs de charge réseau ne prennent pas en charge le routage basé sur le contenu, le trafic de test doit être envoyé vers un port différent du trafic de production.

Écouteurs et règles

Pour les blue/green déploiements avec un Network Load Balancer, vous devez configurer des écouteurs :

- Écouteur de production : gère le trafic de production (généralement sur le port 80 ou 443)
  - Transfère initialement le trafic au groupe cible principal (révision du service bleu)
  - Après le déploiement, transfert le trafic vers le groupe cible alternatif (révision du service écologique)
- Écouteur de test (facultatif) : gère le trafic de test pour valider la révision du service écologique avant de transférer le trafic de production
  - Peut être configuré sur un port différent (par exemple, 8080 ou 8443)
  - Transfère le trafic vers le groupe cible alternatif (révision du service écologique) pendant les tests

Contrairement aux équilibreurs de charge d'application, les équilibreurs de charge réseau ne prennent pas en charge les règles de routage basées sur le contenu. Au lieu de cela, le trafic est acheminé en fonction du port et du protocole de l'écouteur.

Les commandes AWS CLI suivantes créent des écouteurs de production et de test pour un Network Load Balancer :

Remplacez le *user-input* par vos valeurs.

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
net/my-network-lb/1234567890123456 \
    --protocol TCP \
    --port 80 \
    --default-actions Type=forward,
 TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/blue-
target-group/1234567890123456

aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
net/my-network-lb/1234567890123456 \
    --protocol TCP \
    --port 8080 \
```

```
    --default-actions Type=forward,
 TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/green-
target-group/1234567890123456
```

Configuration du service

Vous devez disposer des autorisations nécessaires pour autoriser Amazon ECS à gérer les ressources de l'équilibreur de charge dans vos clusters en votre nom. Pour de plus amples informations, veuillez consulter Rôle IAM de l'infrastructure Amazon ECS pour les équilibreurs de charge.

Lorsque vous créez ou mettez à jour un service Amazon ECS pour blue/green des déploiements avec un Network Load Balancer, vous devez spécifier la configuration suivante :

Remplacez le *user-input* par vos valeurs.

Les principaux composants de cette configuration sont les suivants :

- `targetGroupArn`: ARN du groupe cible principal (révision du service bleu)
- `alternateTargetGroupArn`: L'ARN du groupe cible alternatif (révision du service vert)
- `productionListenerArn`: l'ARN de l'écouteur pour le trafic de production
- `testListenerArn`: (Facultatif) L'ARN de l'écouteur pour le trafic de test
- `roleArn`: l'ARN du rôle qui permet à Amazon ECS de gérer les ressources Network Load Balancer
- `strategy`: défini sur pour activer `BLUE_GREEN` les blue/green déploiements
- `bakeTimeInMinutes`: le temps d'attente après le déploiement de la révision du service écologique avant de transférer le trafic de production

```
{
    "loadBalancers": [
        {
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/blue-target-
group/1234567890123456",
            "containerName": "container-name",
            "containerPort": 80,
            "advancedConfiguration": {
```

```
                "alternateTargetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/green-target-
group/1234567890123456",
                "productionListenerArn":
 "arn:aws:elasticloadbalancing:region:123456789012:listener/net/my-network-
lb/1234567890123456/1234567890123456",
                "testListenerArn":
 "arn:aws:elasticloadbalancing:region:123456789012:listener/net/my-network-
lb/1234567890123456/2345678901234567",
                "roleArn": "arn:aws:iam::123456789012:role/ecs-nlb-role"
            }
        }
    ],
    "deploymentConfiguration": {
        "strategy": "BLUE_GREEN",
        "maximumPercent": 200,
        "minimumHealthyPercent": 100,
        "bakeTimeInMinutes": 5
    }
}
```

Flux de trafic pendant le déploiement

Lors d'un blue/green déploiement avec un Network Load Balancer, le trafic circule dans le système comme suit :

1. État initial : tout le trafic de production est acheminé vers le groupe cible principal (révision du service bleu).
2. Déploiement de la révision du service écologique : Amazon ECS déploie les nouvelles tâches et les enregistre auprès du groupe cible alternatif.
3. Trafic de test : si un écouteur de test est configuré, le trafic de test est acheminé vers le groupe cible alternatif pour valider la révision du service vert.
4. Transfert du trafic de production : Amazon ECS met à jour l'écouteur de production pour acheminer le trafic vers le groupe cible alternatif (révision du service vert).
5. Temps de cuisson : durée pendant laquelle les révisions du service bleu et vert sont exécutées simultanément après le déplacement du trafic de production.
6. Achèvement : Après un déploiement réussi, la révision du service bleu est interrompue.

Si des problèmes sont détectés lors du déploiement, Amazon ECS peut automatiquement revenir en arrière en redirigeant le trafic vers le groupe cible principal (révision du service bleu).

Ressources Service Connect pour les déploiements bleu/vert d'Amazon ECS

Lorsque vous utilisez Service Connect avec blue/green des déploiements, vous devez configurer des composants spécifiques pour permettre le routage correct du trafic entre les révisions de service bleues et vertes. Cette section explique les composants requis et leur configuration.

Présentation de l’architecture

Service Connect développe à la fois des fonctionnalités de découverte de services et de maillage de services par le biais d'un proxy parallèle géré qui est automatiquement injecté dans vos tâches Amazon ECS. Ces proxys gèrent les décisions de routage, les nouvelles tentatives et la collecte des métriques, tout en AWS Cloud Map fournissant le backend du registre des services. Lorsque vous déployez un service avec Service Connect activé, le service s'enregistre et les AWS Cloud Map services clients le découvrent via l'espace de noms.

Dans une implémentation standard de Service Connect, les services client se connectent à des noms de service logiques, et le proxy sidecar gère le routage vers les instances de service réelles. Avec blue/green les déploiements, ce modèle est étendu pour inclure le routage du trafic de test via la `testTrafficRules` configuration.

Au cours d'un blue/green déploiement, les composants clés suivants fonctionnent ensemble :

- Proxy Service Connect : tout le trafic entre les services passe par le proxy Service Connect, qui prend les décisions de routage en fonction de la configuration.
- AWS Cloud Map Enregistrement : les déploiements bleu et vert sont enregistrés auprès de AWS Cloud Map, mais le déploiement vert est initialement enregistré en tant que point de terminaison « test ».
- Routage du trafic de test : la `testTrafficRules` configuration de Service Connect détermine comment identifier et acheminer le trafic de test vers le déploiement écologique. Cela se fait par le biais d'un routage basé sur les en-têtes, dans le cadre duquel des en-têtes HTTP spécifiques contenus dans les demandes dirigent le trafic vers la révision de test. Par défaut, Service Connect reconnaît l'`x-amzn-ecs-blue-green-test`en-tête des protocoles HTTP lorsqu'aucune règle personnalisée n'est spécifiée.
- Configuration du client : tous les clients de l'espace de noms reçoivent automatiquement les itinéraires de production et de test, mais seules les demandes correspondant aux règles de test seront transférées vers le déploiement écologique.

La puissance de cette approche réside dans le fait qu'elle permet de gérer la complexité de la découverte de services pendant les transitions. À mesure que le trafic passe du déploiement bleu au déploiement vert, tous les mécanismes de connectivité et de découverte sont automatiquement mis à jour. Il n'est pas nécessaire de mettre à jour les enregistrements DNS, de reconfigurer les équilibreurs de charge ou de déployer les modifications apportées à la découverte des services séparément, car le maillage de services gère tout cela.

Acheminement et tests du trafic

Service Connect fournit des fonctionnalités avancées de routage du trafic pour les blue/green déploiements, notamment le routage basé sur les en-têtes et la configuration d'alias client pour les scénarios de test.

Tester les règles d'en-tête du trafic

Au cours blue/green des déploiements, vous pouvez configurer les règles d'en-tête du trafic de test pour acheminer des demandes spécifiques vers la (nouvelle) révision du service vert à des fins de test. Cela vous permet de valider la nouvelle version avec un trafic contrôlé avant de terminer le déploiement.

Service Connect utilise le routage basé sur les en-têtes pour identifier le trafic de test. Par défaut, Service Connect reconnaît l'`x-amzn-ecs-blue-green-test`en-tête des protocoles HTTP lorsqu'aucune règle personnalisée n'est spécifiée. Lorsque cet en-tête est présent dans une demande, le proxy Service Connect achemine automatiquement la demande vers le déploiement écologique à des fins de test.

Les règles d'en-tête du trafic de test vous permettent de :

- Acheminer les demandes avec des en-têtes spécifiques vers la révision du service vert
- Testez de nouvelles fonctionnalités avec un sous-ensemble de trafic
- Valider le comportement du service avant le transfert complet du trafic
- Mettre en œuvre des stratégies de test Canary
- Réaliser des tests d'intégration dans un environnement similaire à celui de la production

Le mécanisme de routage basé sur les en-têtes fonctionne parfaitement avec l'architecture de votre application existante. Les services clients n'ont pas besoin de connaître le processus de blue/green déploiement : ils incluent simplement les en-têtes appropriés lors de l'envoi de demandes de test, et le proxy Service Connect gère automatiquement la logique de routage.

Pour plus d'informations sur la configuration des règles d'en-tête du trafic de test, consultez ServiceConnectTestTrafficHeaderRulesle manuel Amazon Elastic Container Service API Reference.

Règles de correspondance des en-têtes

Les règles de correspondance des en-têtes définissent les critères de routage du trafic de test lors blue/green des déploiements. Vous pouvez configurer plusieurs conditions de correspondance pour contrôler avec précision quelles demandes sont acheminées vers la révision du service vert.

La correspondance des en-têtes prend en charge :

- Correspondance exacte de la valeur d'en-tête
- Vérification de la présence des en-têtes
- Correspondance basée sur des motifs
- Combinaisons d'en-têtes multiples

Les exemples d'utilisation incluent le routage des demandes avec des chaînes d'agent utilisateur, des versions d'API ou des indicateurs de fonctionnalité spécifiques vers le service vert à des fins de test.

Pour plus d'informations sur la configuration de la correspondance des en-têtes, consultez ServiceConnectTestTrafficHeaderMatchRulesle manuel Amazon Elastic Container Service API Reference.

Alias clients pour les déploiements blue/green

Les alias de client fournissent des points de terminaison DNS stables pour les services lors blue/green des déploiements. Ils permettent un routage fluide du trafic entre les révisions de service bleues et vertes sans obliger les applications clientes à modifier leurs points de terminaison de connexion.

Lors d'un blue/green déploiement, les alias du client :

- Maintenir des noms DNS cohérents pour les connexions client
- Activer la commutation automatique du trafic entre les révisions de service
- Support des stratégies de migration progressive du trafic
- Offrez des fonctionnalités de restauration en redirigeant le trafic vers la version bleue

Vous pouvez configurer plusieurs alias de client pour différents ports ou protocoles, ce qui permet à des architectures de service complexes de maintenir la connectivité pendant les déploiements.

Pour plus d'informations sur la configuration des alias clients, consultez le ServiceConnectClientAliasmanuel Amazon Elastic Container Service API Reference.

Bonnes pratiques pour le routage du trafic

Lors de la mise en œuvre du routage du trafic pour blue/green les déploiements avec Service Connect, tenez compte des meilleures pratiques suivantes :

- Commencez par des tests basés sur les en-têtes : utilisez les règles d'en-tête du trafic de test pour valider le service écologique avec un trafic contrôlé avant de changer l'ensemble du trafic.
- Configurer les contrôles de santé : assurez-vous que les services bleu et vert disposent de contrôles de santé appropriés configurés pour empêcher le routage du trafic vers des instances défectueuses.
- Surveillez les indicateurs de service : suivez les indicateurs de performance clés pour les deux révisions de service pendant le déploiement afin d'identifier les problèmes à un stade précoce.
- Planifier une stratégie de restauration : configurez les alias des clients et les règles de routage pour permettre un retour rapide au service bleu si des problèmes sont détectés.
- Testez la logique de correspondance des en-têtes : validez vos règles de correspondance des en-têtes dans un environnement hors production avant de les appliquer aux déploiements de production.

Flux de travail de blue/green déploiement de Service Connect

Comprendre comment Service Connect gère le processus de blue/green déploiement vous permet de mettre en œuvre et de dépanner efficacement vos déploiements. Le flux de travail suivant montre comment les différents composants interagissent au cours de chaque phase du déploiement.

Phases de déploiement

Le blue/green déploiement de Service Connect passe par plusieurs phases distinctes :

1. État initial : Le service bleu gère 100 % du trafic de production. Tous les services clients de l'espace de noms se connectent au service bleu via le nom de service logique configuré dans Service Connect.
2. Enregistrement du service écologique : Lorsque le déploiement écologique commence, il est enregistré en AWS Cloud Map tant que point de terminaison « test ». Le proxy Service Connect des services clients reçoit automatiquement les configurations d'itinéraires de production et de test.

3. Routage du trafic de test : les demandes contenant les en-têtes de trafic de test (tels que `x-amzn-ecs-blue-green-test`) sont automatiquement acheminées vers le service vert par le proxy Service Connect. Le trafic de production continue d'affluer vers le service bleu.
4. Préparation du transfert du trafic : une fois les tests réussis, le processus de déploiement prépare le transfert du trafic de production. Les services bleus et verts restent enregistrés et sains.
5. Transfert du trafic de production : la configuration de Service Connect est mise à jour pour acheminer le trafic de production vers le service écologique. Cela se produit automatiquement sans nécessiter de mises à jour du service client ou de modifications du DNS.
6. Période de cuisson : durée pendant laquelle les révisions du service bleu et vert sont exécutées simultanément après le déplacement du trafic de production.
7. Annulation de l'enregistrement du service bleu : une fois le transfert de trafic et la validation réussis, le service bleu est désenregistré AWS Cloud Map et le déploiement est terminé.

Comportement du proxy Service Connect

Le proxy Service Connect joue un rôle crucial dans la gestion du trafic lors des blue/green déploiements. Comprendre son comportement vous aide à concevoir des stratégies de test et de déploiement efficaces.

Principaux comportements des proxys lors des blue/green déploiements :

- Découverte automatique des itinéraires : le proxy découvre automatiquement les itinéraires de production et de test AWS Cloud Map sans qu'il soit nécessaire de redémarrer l'application ou de modifier la configuration.
- Routage basé sur les en-têtes : le proxy examine les en-têtes des demandes entrantes et achemine le trafic vers la version de service appropriée en fonction des règles de trafic de test configurées.
- Intégration du contrôle de santé : le proxy achemine le trafic uniquement vers des instances de service saines, excluant automatiquement les tâches défectueuses du pool de routage.
- Réessai et interruption de circuit : le proxy intègre une logique de réessai et des fonctionnalités de coupure de circuit, améliorant ainsi la résilience lors des déploiements.
- Collecte de métriques : le proxy collecte des métriques détaillées pour les services bleus et verts, permettant une surveillance complète lors des déploiements.

Mises à jour de découverte de

L'un des principaux avantages de l'utilisation de Service Connect pour les blue/green déploiements est la gestion automatique des mises à jour de découverte de services. Les blue/green déploiements traditionnels nécessitent souvent des mises à jour DNS complexes ou une reconfiguration de l'équilibreur de charge, mais Service Connect gère ces modifications de manière transparente.

Au cours d'un déploiement, Service Connect gère :

- Mises à jour de l'espace de noms : L'espace de noms Service Connect inclut automatiquement les points de terminaison de service bleus et verts, avec des règles de routage appropriées.
- Configuration du client : tous les services clients de l'espace de noms reçoivent automatiquement des informations de routage mises à jour sans qu'il soit nécessaire de redémarrer ou de redéployer.
- Transition progressive : les mises à jour de découverte des services se font progressivement et en toute sécurité, sans perturber les demandes en cours.
- Support d'annulation : si une annulation est nécessaire, Service Connect peut rapidement rétablir les configurations de découverte de services afin de rediriger le trafic vers le service bleu.

Création d'un blue/green déploiement Amazon ECS

En utilisant les blue/green déploiements Amazon ECS, vous pouvez apporter et tester des modifications de service avant de les implémenter dans un environnement de production.

Prérequis

Effectuez les opérations suivantes avant de démarrer un blue/green déploiement.

1. Configurez les autorisations appropriées.
   - Pour plus d'informations sur les autorisations d'Elastic Load Balancing, consultezRôle IAM de l'infrastructure Amazon ECS pour les équilibreurs de charge.
   - Pour plus d'informations sur les autorisations Lambda, voir Autorisations requises pour les fonctions Lambda dans les déploiements Amazon ECS blue/green
2. Les blue/green déploiements Amazon ECS nécessitent que votre service utilise l'une des fonctionnalités suivantes : Configurez les ressources appropriées.
   - Application Load Balancer - Pour plus d'informations, consultez. Ressources d'Application Load Balancer pour les déploiements blue/green

- Network Load Balancer - Pour plus d'informations, consultez. Ressources Network Load Balancer pour les déploiements bleu/vert d'Amazon ECS
- Service Connect - Pour plus d'informations, consultezRessources Service Connect pour les déploiements bleu/vert d'Amazon ECS.

3. Créez une règle pour acheminer le trafic vers votre version de service écologique. Pour plus d'informations, consultez les règles relatives aux écouteurs dans le guide de l'utilisateur du Network Load Balancer.
4. Créez un groupe cible pour la révision de votre service écologique. Lorsque vous utilisez le mode `awsvpc` réseau pour vos tâches, le type de cible doit être`ip`. Pour plus d'informations sur les groupes cibles, consultez la section Groupes cibles dans le guide de l'utilisateur du Network Load Balancer.
5. Décidez si vous souhaitez exécuter des fonctions Lambda pour les événements du cycle de vie.
   - Avant la mise à l'échelle
   - Après la mise à l'échelle
   - Testez le transfert de trafic
   - Après le test, changement de trafic
   - Réorientation du trafic de production
   - Après le transfert du trafic de production

   Créez des fonctions Lambda pour chaque événement du cycle de vie. Pour plus d'informations, voir Création d'une fonction Lambda avec la console dans le Guide du AWS Lambda développeur.

Procédure

Vous pouvez utiliser la console ou le AWS CLI pour créer un blue/green service Amazon ECS.

Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Déterminez la ressource à partir de laquelle vous lancez le service.

Pour démarrer un service à partir de	Étapes	
Clusters	a. Sur la page Clusters, sélectionnez le cluster dans lequel créer le service. La page de détails du cluster s'affiche. b. Dans l'onglet Services choisissez Create (Créer).	
Définition de tâche	a. Sur la page Définitions de tâches, sélectionnez la définition de tâche. b. Dans le menu Déployer, choisissez Create service.	

La page Créer un service s'affiche.

3. Sous Détails du service, procédez comme suit :

   a. Pour Famille de définitions de tâches, choisissez la définition de tâche à utiliser. Ensuite, pour Révision de la définition des tâches, entrez la révision à utiliser.

   b. Pour Service name (Nom du service), saisissez un nom pour votre service.

4. Pour exécuter le service dans un cluster existant, dans Cluster existant, sélectionnez le cluster. Pour exécuter le service dans un nouveau cluster, choisissez Create cluster

5. Choisissez la manière dont vos tâches sont réparties dans l'infrastructure de votre cluster. Sous Configuration informatique, choisissez votre option.

Option de calcul	Étapes	
Stratégie de fournisseur de capacité	a. Sous Options de calcul, choisissez Stratégie de fournisseur de capacité. b. Choisissez une stratégie : • Pour utiliser une stratégie de fournisseur de capacité, choisissez Use cluster default (Utiliser la stratégie par défaut du cluster). • Si votre cluster ne possède pas de stratégie de fournisseur de capacité par défaut ou si vous souhaitez utiliser une stratégie personnalisée, sélectionnez Utiliser une stratégie personnalisée ou Ajouter une stratégie de fournisseur de capacité et définissez votre stratégie de fournisseur de capacité personnalisée en spécifiant une Base, un Fournisseur de capacité et un Poids.	

Option de calcul	Étapes
	Note Pour utiliser un fournisseur de capacité dans une stratégie, le fournisseur de capacité doit être associé au cluster.
Type de lancement	a. Dans Compute options (Options de calcul), sélectionnez Launch type (Type de lancement). b. Pour Launch type (Type de lancement), choisissez un type de lancement. c. (Facultatif) Lorsque le type de lancement Fargate est spécifié, pour Platform version (Version de plateforme), spécifiez la version de plateforme à utiliser. Si vous ne spécifiez aucune version de plateforme, la version `LATEST` est utilisée.

6. Sous Configuration du déploiement, procédez comme suit :

   a. Pour le type de service, choisissez Replica.

   b. Pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer et à conserver dans le service.

c. Pour qu'Amazon ECS surveille la répartition des tâches entre les zones de disponibilité et les redistribue en cas de déséquilibre, sous Rééquilibrage du service des zones de disponibilité, sélectionnez Rééquilibrage du service des zones de disponibilité.

d. Pour la période de grâce du bilan de santé, entrez la durée (en secondes) pendant laquelle le planificateur de services ignore les vérifications d'état incorrectes d'Elastic Load Balancing, de VPC Lattice et de l'état des conteneurs après le premier démarrage d'une tâche. Si vous ne spécifiez aucune valeur de période de grâce pour le bilan de santé, la valeur par défaut de 0 est utilisée.

7. a. Pour le temps de cuisson, entrez le nombre de minutes pendant lesquelles les révisions du service bleu et vert seront exécutées simultanément avant que la révision bleue ne soit terminée. Cela laisse le temps de procéder à des vérifications et à des tests.

   b. (Facultatif) Exécutez les fonctions Lambda à exécuter à des étapes spécifiques du déploiement. Sous Hooks du cycle de vie du déploiement, sélectionnez les étapes pour exécuter les hooks du cycle de vie.

      Pour ajouter un crochet de cycle de vie :

      a. Choisissez Ajouter.

      b. Pour la fonction Lambda, entrez le nom ou l'ARN de la fonction.

      c. Pour Rôle, sélectionnez le rôle IAM autorisé à appeler la fonction Lambda.

      d. Pour les étapes du cycle de vie, sélectionnez les étapes pendant lesquelles la fonction Lambda doit être exécutée.

8. Pour configurer la manière dont Amazon ECS détecte et gère les échecs de déploiement, développez Deployment failure detection (Détection des échecs de déploiement), puis choisissez vos options.

   a. Pour arrêter un déploiement lorsque les tâches ne peuvent pas démarrer, sélectionnez Use the Amazon ECS deployment circuit breaker (Utiliser le disjoncteur de déploiement Amazon ECS).

      Pour que le logiciel annule automatiquement le déploiement au dernier état de déploiement terminé lorsque le disjoncteur de déploiement met le déploiement en état d'échec, sélectionnez Annulation en cas d'échec.

   b. Pour arrêter un déploiement en fonction des métriques de l'application, sélectionnez Utiliser une ou plusieurs CloudWatch alarmes. Ensuite, à partir du nom de l'CloudWatch

alarme, choisissez les alarmes. Pour créer une nouvelle alarme, rendez-vous sur la CloudWatch console.

Pour que le logiciel annule automatiquement le déploiement au dernier état de déploiement terminé lorsqu'une CloudWatch alarme indique que le déploiement a échoué, sélectionnez Annulation en cas d'échec.

9. (Facultatif) Pour interconnecter votre service à l'aide de Service Connect, étendez Service Connect, puis spécifiez les éléments suivants :

   a. Sélectionnez Activer Service Connect.

   b. Sous Service Connect configuration (Configuration de Service Connect), spécifiez le mode client.

      - Si votre service exécute une application client réseau qui doit uniquement se connecter aux autres services de l'espace de noms, choisissez Client side only.
      - Si votre service exécute une application réseau ou un service Web et doit fournir des points de terminaison pour ce service, et se connecte à d'autres services dans l'espace de noms, choisissez Client and server (Client et serveur).

   c. Pour utiliser un espace de noms qui n'est pas l'espace de noms de cluster par défaut, dans Namespace (Espace de noms), choisissez l'espace de noms du service.

   d. (Facultatif) Configurez les règles d'en-tête du trafic de test pour les blue/green déploiements. Sous Tester le routage du trafic, spécifiez les éléments suivants :

      i. Sélectionnez Activer les règles d'en-tête du trafic de test pour acheminer des demandes spécifiques vers la révision du service vert pendant les tests.

      ii. Pour les règles de correspondance des en-têtes, configurez les critères de routage du trafic de test :

         - Nom de l'en-tête : entrez le nom de l'en-tête HTTP correspondant (par exemple, `X-Test-Version` ou`User-Agent`).
         - Type de correspondance : Choisissez les critères de correspondance :
            - Correspondance exacte : demandes de routage dont la valeur d'en-tête correspond exactement à la valeur spécifiée
            - En-tête présent : Router les demandes contenant l'en-tête spécifié, quelle que soit sa valeur

- Correspondance de modèle : demandes de routage dont la valeur d'en-tête correspond à un modèle spécifié
- Valeur d'en-tête (si vous utilisez une correspondance exacte ou une correspondance de modèle) : entrez la valeur ou le modèle à comparer.

Vous pouvez ajouter plusieurs règles de correspondance d'en-têtes pour créer une logique de routage complexe. Les demandes correspondant à l'une des règles configurées seront acheminées vers la version verte du service à des fins de test.

iii. Choisissez Ajouter une règle d'en-tête pour configurer des conditions de correspondance d'en-tête supplémentaires.

> **Note**
>
> Les règles d'en-tête du trafic de test vous permettent de valider les nouvelles fonctionnalités avec un trafic contrôlé avant de terminer le déploiement complet. Cela vous permet de tester la révision du service vert avec des demandes spécifiques (telles que celles provenant d'outils de test internes ou d'utilisateurs bêta) tout en maintenant un flux de trafic normal vers la révision du service bleu.

e. (Facultatif) Spécifiez une configuration de journal. Sélectionnez Utiliser la collecte de journaux. L'option par défaut envoie les journaux du conteneur à CloudWatch Logs. Les autres options du pilote de journal sont configurées à l'aide de AWS FireLens. Pour de plus amples informations, veuillez consulter Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

Voici une description plus détaillée de chaque destination de journal de conteneur.

- Amazon CloudWatch — Configurez la tâche pour envoyer les journaux des conteneurs à CloudWatch Logs. Les options du pilote de journal par défaut sont fournies, ce qui permet de créer un groupe de CloudWatch journaux en votre nom. Pour spécifier un autre nom de groupe de journaux, modifiez les valeurs des options de pilote.
- Amazon Data Firehose : configurez la tâche pour envoyer les journaux des conteneurs à Firehose. Les options du pilote de journal par défaut sont fournies, qui envoient les journaux à un flux de diffusion Firehose. Pour spécifier un autre nom de flux de diffusion, modifiez les valeurs des options de pilote.

- Amazon Kinesis Data Streams : configurez la tâche pour envoyer les journaux des conteneurs à Kinesis Data Streams. Les options du pilote de journal par défaut sont fournies, qui envoient les journaux vers un flux Kinesis Data Streams. Pour spécifier un autre nom de flux, modifiez les valeurs des options de pilote.
- Amazon OpenSearch Service — Configurez la tâche pour envoyer les journaux des conteneurs vers un domaine OpenSearch de service. Les options de pilote de journal doivent être fournies.
- Amazon S3 — Configurez la tâche pour envoyer les journaux des conteneurs vers un compartiment Amazon S3. Les options du pilote de journal par défaut sont fournies, mais vous devez spécifier un nom de compartiment Amazon S3 valide.

10. (Facultatif) Configurez l'équilibrage de charge pour un déploiement bleu/vert.

Type d'Elastic Load Balancing	Étapes	
Application Load Balancer	a. Pour le type d'équilibreur de charge, choisissez Application Load Balancer. b. Choisissez Create a new load balancer (Créer un équilibreur de charge) pour créer un équilibreur de charge Application Load Balancer ou Use an existing load balancer (Utiliser un équilibreur de charge existant) pour sélectionner un équilibreur de charge Application Load Balancer existant. c. Pour Container, choisissez le conteneur qui héberge le service.	

Type d'Elastic Load Balancing	Étapes	
	d. Pour Load balancer name (nom d'équilibreur de charge), saisissez un nom unique. e. Pour Listener (Écouteur), saisissez un port et un protocole sur lesquels Application Load Balancer doit écouter les demandes de connexion . Par défaut, l'équilibreur de charge est configuré pour utiliser le port 80 et HTTP. • Pour Règle de production, entrez l'ordre d'évaluation et le modèle de chemin de la règle. Cette règle concerne le trafic de révision de votre service de production (bleu). • Pour Règle de test, entrez l'ordre d'évaluat ion et le modèle de chemin de la règle. Cette règle concerne le trafic de révision de votre service de test (vert).	

Type d'Elastic Load Balancing	Étapes	
	f. Pour le groupe cible, configurez les éléments suivants : • Pour Target group name (Nom du groupe cible), saisissez un nom et un protocole pour le groupe cible vers lequel Application Load Balancer achemine les requêtes. • Pour Protocol, choisissez le protocole pour le groupe cible vers lequel l'Application Load Balancer achemine les demandes. Par défaut, le groupe cible achemine les requêtes vers le premier conteneur défini dans votre définition de tâche. • Pour le délai de désenregistrement, entrez le nombre de secondes pendant lequel l'équilibreur de charge doit passer à l'état cible. `UNUSED` La durée par défaut est 300 secondes.	

Type d'Elastic Load Balancing	Étapes	
	• Pour Health check path (Chemin de surveillance de l'état), saisissez un chemin qui existe dans votre conteneur où Application Load Balancer doit envoyer périodiquement des requêtes pour vérifier l'état de la connexion entre Application Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (/). • Dans Nom de groupe alternatif, entrez le nom du groupe cible pour votre révision de service de test (vert).	

Type d'Elastic Load Balancing	Étapes	
Network Load Balancer	a. Pour Load balancer type (Type d'équilibreur de charge), sélectionnez Network Load Balancer. b. Dans le champ Load Balancer (Équilibreur de charge), choisissez Attacher à un Network Load Balancer existant. c. Pour Choose container to load balance (Choisissez le conteneur pour équilibrer la charge), choisissez le conteneur qui héberge le service. d. Pour Écouteur de production, choisissez le port d'écouteur de production et le protocole d'écouteur de production. Il s'agit de l'écouteur pour le trafic de révision de votre service de production (bleu). e. Pour Test listener, choisissez le port Test listener et le protocole Test listener. Il s'agit de l'écouteur pour le trafic de révision	

Type d'Elastic Load Balancing	Étapes	
	de votre service de test (vert). f. Pour le groupe cible, configurez les éléments suivants : • Pour Target group name (Nom du groupe cible), saisissez un nom et un protocole pour le groupe cible vers lequel Network Load Balancer achemine les requêtes. • Pour Protocol, choisissez le protocole pour le groupe cible vers lequel le Network Load Balancer achemine les demandes. Par défaut, le groupe cible achemine les requêtes vers le premier conteneur défini dans votre définition de tâche. • Pour le délai de désenregistrement, entrez le nombre de secondes pendant lequel l'équilibreur de charge doit passer à l'état cible. `UNUSED` La	

Type d'Elastic Load Balancing	Étapes	
	durée par défaut est 300 secondes. • Pour Health check path (Chemin de surveillance de l'état), saisissez un chemin qui existe dans votre conteneur où Application Load Balancer doit envoyer périodiquement des requêtes pour vérifier l'état de la connexion entre Application Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (/). • Dans Nom de groupe alternatif, entrez le nom du groupe cible pour votre révision de service de test (vert).	

11. (Facultatif) Pour vous aider à identifier votre service et vos tâches, développez Tags (balises), puis configurez vos balises.

    Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises de définition des tâches, sélectionnez Activer les balises gérées Amazon ECS, puis pour Propager des balises à partir de, choisissez Définitions de tâches.

Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises d'un service, sélectionnez Activer les balises gérées Amazon ECS, puis pour Propager des balises à partir de, choisissez Service.

Ajoutez ou supprimez une balise.

- [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :
  - Pour Clé, saisissez le nom de la clé.
  - Pour Valeur, saisissez la valeur de clé.
- [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.

12. Choisissez Créer.

AWS CLI

1. Créez un fichier nommé `service-definition.json` avec le contenu suivant.

   Remplacez le *`user-input`* par vos valeurs.

```
{
  "serviceName": "myBlueGreenService",
  "cluster": "arn:aws:ecs:us-west-2:123456789012:cluster/sample-fargate-
cluster",
  "taskDefinition": "sample-fargate:1",
  "desiredCount": 5,
  "launchType": "FARGATE",
  "networkConfiguration": {
    "awsvpcConfiguration": {
      "subnets": [
        "subnet-09ce6e74c116a2299",
        "subnet-00bb3bd7a73526788",
        "subnet-0048a611aaec65477"
      ],
      "securityGroups": [
        "sg-09d45005497daa123"
      ],
      "assignPublicIp": "ENABLED"
    }
  },
  "deploymentController": {
    "type": "ECS"
```

```
  },
  "deploymentConfiguration": {
    "strategy": "BLUE_GREEN",
    "maximumPercent": 200,
    "minimumHealthyPercent": 100,
    "bakeTimeInMinutes": 2,
    "alarms": {
      "alarmNames": [
        "myAlarm"
      ],
      "rollback": true,
      "enable": true
    },
    "lifecycleHooks": [
      {
        "hookTargetArn": "arn:aws:lambda:us-
west-2:7123456789012:function:checkExample",
        "roleArn": "arn:aws:iam::123456789012:role/ECSLifecycleHookInvoke",
        "lifecycleStages": [
          "PRE_SCALE_UP"
        ]
      }
    ]
  },
  "loadBalancers": [
    {
      "targetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/blue-target-group/54402ff563af1197",
      "containerName": "fargate-app",
      "containerPort": 80,
      "advancedConfiguration": {
        "alternateTargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/green-target-group/cad10a56f5843199",
        "productionListenerRule": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener-rule/app/my-blue-green-
demo/32e0e4f946c3c05b/9cfa8c482e204f7d/831dbaf72edb911",
        "roleArn": "arn:aws:iam::123456789012:role/LoadBalancerManagementforECS"
      }
    }
  ]
}
```

2. Exécutez `create-service`.

Remplacez le *user-input* par vos valeurs.

```
aws ecs create-service --cli-input-json file://service-definition.json
```

Vous pouvez également utiliser l'exemple suivant qui crée un service de blue/green déploiement avec une configuration d'équilibreur de charge :

```
aws ecs create-service \
   --cluster "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster" \
   --service-name "blue-green-example-service" \
   --task-definition "nginxServer:1" \
   --launch-type "FARGATE" \
   --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12345,subnet-67890,subnet-abcdef,subnet-
fedcba],securityGroups=[sg-12345],assignPublicIp=ENABLED}" \
   --desired-count 3 \
   --deployment-controller "type=ECS" \
   --deployment-configuration
 "strategy=BLUE_GREEN,maximumPercent=200,minimumHealthyPercent=100,bakeTimeInMinutes=0"
 \
   --load-balancers "targetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/MyBGtg1/
abcdef1234567890,containerName=nginx,containerPort=80,advancedConfiguration={alternateTa
west-2:123456789012:targetgroup/
MyBGtg2/0987654321fedcba,productionListenerRule=arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener-rule/app/
MyLB/1234567890abcdef/1234567890abcdef,roleArn=arn:aws:iam::123456789012:role/
ELBManagementRole}"
```

Étapes suivantes

- Mettez à jour le service pour démarrer le déploiement. Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.
- Surveillez le processus de déploiement pour vous assurer qu'il suit le blue/green modèle :
  - La révision du service écologique est créée et étendue
  - Le trafic de test est acheminé vers la version verte (si elle est configurée)
  - Le trafic de production passe à la version verte
  - Après le temps de cuisson, la révision bleue est terminée

Déployez les services Amazon ECS à l'aide d'un contrôleur tiers

Le type de déploiement externe vous permet d'utiliser n'importe quel contrôleur de déploiement tiers pour un contrôle complet du processus de déploiement pour un service Amazon ECS service. Les détails pour votre service sont gérés par les actions d'API de gestion de service (`CreateService`, `UpdateService` et `DeleteService`) ou par les actions d'API de gestion des tâches définies (`CreateTaskSet`, `UpdateTaskSet`, `UpdateServicePrimaryTaskSet` et `DeleteTaskSet`). Chaque action d'API gère un sous-ensemble de paramètres de définition de service.

L'action d'API `UpdateService` met à jour le nombre souhaité et les paramètres de période de délai supplémentaire des surveillances de l'état pour un service. Si le type de lancement, la version de plateforme, les détails de l'équilibreur de charge, la configuration du réseau ou la définition de tâche doivent être mis à jour, vous devez créer une nouvelle tâche.

L'action d'API `UpdateTaskSet` met à jour uniquement le paramètre d'échelle pour un ensemble de tâches.

L'action d'API `UpdateServicePrimaryTaskSet` modifie l'ensemble de tâches dans un service qui est l'ensemble de tâches principal. Lorsque vous appelez l'action d'API `DescribeServices`, elle renvoie tous les champs spécifiés pour un ensemble de tâches principal. Si l'ensemble de tâches principal d'un service est mis à jour, toutes les valeurs de paramètres de l'ensemble de tâches qui existe sur l'ensemble de tâches principal différent de l'ancienne tâche principale dans un service sont mises à jour vers la nouvelle valeur lorsqu'un ensemble de tâches principal est défini. Si aucun ensemble de tâches principale n'est défini pour un service, les champs de l'ensemble de tâches sont vides pour la description de service.

Considérations relatives au déploiement externe

Tenez compte des éléments suivants lorsque vous utilisez le type de déploiement externe :

- Les types d'équilibreur de charge pris en charge sont un Application Load Balancer ou un Network Load Balancer.
- Le type de lancement Fargate ou les types de contrôleur de déploiement `EXTERNAL` ne prennent pas en charge la stratégie de planification `DAEMON`.
- L' AutoScaling intégration des applications avec Amazon ECS prend uniquement en charge le service Amazon ECS en tant que cible. La dimension évolutive prise en charge pour Amazon ECS est `ecs:service:DesiredCount` le nombre de tâches d'un service Amazon ECS. Il n'existe aucune intégration directe entre l'application AutoScaling et les ensembles de tâches Amazon

ECS. Les ensembles de tâches Amazon ECS calculent le en `ComputedDesiredCount` fonction du service Amazon ECS`DesiredCount`.

Workflow de déploiement externe

Voici le flux de travail de base pour gérer un déploiement externe sur Amazon ECS.

Pour gérer un service Amazon ECS service à l'aide d'un contrôleur de déploiement externe

1. Créez un service Amazon ECS service. Le seul paramètre requis est le nom de service. Vous pouvez spécifier les paramètres suivants lors de la création d'un service à l'aide d'un contrôleur de déploiement externe. Tous les autres paramètres de service sont spécifiés lors de la création d'un ensemble de tâches dans le service.

   `serviceName`

   Type : String

   Obligatoire : oui

   Nom de votre service. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres, des tirets ou des traits de soulignement. Dans un cluster, les noms de service doivent être uniques, mais certains services peuvent avoir des noms similaires dans des clusters différents d'une même région ou de plusieurs régions.

   `desiredCount`

   Le nombre d'instances sur la définition de tâche d'ensemble de tâches spécifié à placer et à continuer d'exécuter au sein du service.

   `deploymentConfiguration`

   Paramètres de déploiement facultatifs qui contrôlent le nombre de tâches exécutées durant le déploiement et la commande d'arrêt et de démarrage des tâches.

   `tags`

   Type : tableau d'objets

   Obligatoire : non

   Métadonnées que vous appliquez au service pour faciliter le classement et l'organisation. Chaque étiquette est constituée d'une clé et d'une valeur facultative que vous définissez.

Lorsqu'un service est supprimé, les étiquettes sont également supprimées. Un maximum de 50 balises peut être appliqué au service. Pour de plus amples informations, veuillez consulter Balisage des ressources Amazon ECS.

`key`

Type : String

Contraintes de longueur : Longueur minimum de 1. Longueur maximale de 128.

Obligatoire : non

Partie d'une paire clé-valeur qui constitue une étiquette. Une clé est une étiquette générale qui fait office de catégorie pour les valeurs d'étiquette plus spécifiques.

`value`

Type : String

Contraintes de longueur : Longueur minimum de 0. Longueur maximale de 256.

Obligatoire : non

Partie facultative d'une paire clé-valeur qui constitue une étiquette. Une valeur agit comme un descripteur au sein d'une catégorie d'étiquette (clé).

`enableECSManagedTags`

Spécifie si utiliser les identifications gérées par Amazon ECS pour les tâches dans le service. Pour de plus amples informations, veuillez consulter Utiliser des tags pour la facturation.

`propagateTags`

Type : String

Valeurs valides : `TASK_DEFINITION` | `SERVICE`

Obligatoire : non

Indique s'il faut copier les étiquettes de la définition de tâche ou du service dans les tâches du service. Si aucune valeur n'est spécifiée, les étiquettes ne sont pas copiées. Les étiquettes ne peuvent être copiées dans les tâches du service que lors de la création du service. Pour ajouter des balises à une tâche après la création du service ou de la tâche, utilisez l'action d'API `TagResource`.

`schedulingStrategy`

Stratégie de planification à utiliser. Les services utilisant un contrôleur de déploiement externe prennent uniquement en charge la stratégie de planification `REPLICA`.

`placementConstraints`

Tableau d'objets de contraintes de placement à utiliser pour les tâches de votre service. Vous pouvez spécifier un maximum de 10 contraintes par tâche (cette limite comprend les contraintes de la définition de tâche et celles spécifiées lors de l'exécution). Si vous utilisez le type de lancement Fargate, les contraintes de placement des tâches ne sont pas prises en charge.

`placementStrategy`

Objets de stratégie de placement à utiliser pour les tâches de votre service. Vous pouvez spécifier jusqu'à quatre règles de stratégie par service.

Voici un exemple de définition de service pour la création d'un service qui utilise un contrôleur de déploiement externe.

```
{
    "cluster": "",
    "serviceName": "",
    "desiredCount": 0,
    "role": "",
    "deploymentConfiguration": {
        "maximumPercent": 0,
        "minimumHealthyPercent": 0
    },
    "placementConstraints": [
        {
            "type": "distinctInstance",
            "expression": ""
        }
    ],
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
```

```
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION"
}
```

2. Créez un ensemble de tâches initial. L'ensemble de tâches contient les détails suivants pour votre service :

   `taskDefinition`

   La définition de tâche pour les tâches dans l'ensemble de tâches à utiliser.

   `launchType`

   Type : String

   Valeurs valides : `EC2` | `FARGATE` | `EXTERNAL`

   Obligatoire : non

   Type de lancement sur lequel vous pouvez exécuter votre service. Si aucun type de lancement n'est spécifié, the `capacityProviderStrategy` par défaut est utilisé par défaut. Pour de plus amples informations, veuillez consulter Types de lancement Amazon ECS.

   Si un `launchType` est spécifié, le paramètre `capacityProviderStrategy` doit être omis.

   `platformVersion`

   Type : chaîne

   Obligatoire : non

Version de la plateforme sur laquelle s'exécutent vos tâches dans le service. Une version de plateforme est spécifiée uniquement pour les tâches utilisant le type de lancement Fargate. Si vous ne spécifiez aucune valeur, la dernière version (`LATEST`) est utilisée par défaut.

AWS Les versions de la plate-forme Fargate sont utilisées pour faire référence à un environnement d'exécution spécifique pour l'infrastructure de tâches Fargate. Lorsque vous spécifiez `LATEST` comme version de plateforme lors de l'exécution d'une tâche ou de la création d'un service, vous obtenez la version de plateforme la plus récente pour vos tâches. Lorsque vous mettez à l'échelle votre service, ces tâches reçoivent la version de plateforme spécifiée dans le déploiement actuel du service. Pour de plus amples informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.

> **Note**
>
> Les versions de plateforme ne sont pas spécifiées pour les tâches utilisant le type de lancement EC2.

`loadBalancers`

Un objet d'équilibreur de charge qui représente l'équilibreur de charge à utiliser avec votre service. Lorsque vous utilisez un contrôleur de déploiement externe, seuls les équilibreurs de charge Application Load Balancer et les Network Load Balancer sont pris en charge. Si vous utilisez un équilibreur de charge Application Load Balancer, seul un groupe cible d'équilibreur de charge Application Load Balancer est autorisé par tâche.

L'extrait de code suivant montre un exemple d'objet `loadBalancer` à utiliser.

```
"loadBalancers": [
        {
            "targetGroupArn": "",
            "containerName": "",
            "containerPort": 0
        }
]
```

> **Note**
>
> Lorsque vous spécifiez un objet `loadBalancer`, vous devez spécifier `targetGroupArn` et omettre les paramètres `loadBalancerName`.

`networkConfiguration`

Configuration réseau du service. Ce paramètre est obligatoire pour les définitions de tâches qui utilisent le mode réseau `awsvpc` afin de recevoir leur propre interface réseau Elastic. Il n'est pas pris en charge avec les autres modes réseau. Pour plus d'informations sur la mise en réseau pour le type de lancement Fargate, consultez. Options de mise en réseau des tâches Amazon ECS pour le type de lancement Fargate

`serviceRegistries`

Les détails des enregistrements de découverte de service à attribuer à ce service. Pour de plus amples informations, veuillez consulter Utilisez la découverte des services pour connecter les services Amazon ECS aux noms DNS.

`scale`

Pourcentage à virgule flottante du nombre souhaité de tâches à placer et à continuer d'exécuter dans l'ensemble de tâches. La valeur est spécifiée en tant que pourcentage total de `desiredCount` d'un service. Les valeurs acceptées sont des nombres compris entre 0 et 100.

Voici un exemple JSON pour créer un ensemble de tâches pour un contrôleur de déploiement externe.

```
{
    "service": "",
    "cluster": "",
    "externalId": "",
    "taskDefinition": "",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
```

```
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "launchType": "EC2",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "scale": {
        "value": null,
        "unit": "PERCENT"
    },
    "clientToken": ""
}
```

3. Lorsque des modifications de service sont nécessaires, utilisez l'action d'API `CreateTaskSet`, `UpdateService` ou `UpdateTaskSet`, en fonction des paramètres que vous mettez à jour. Si vous avez créé un ensemble de tâches, utilisez le paramètre `scale` pour chaque ensemble de tâches dans un service afin de déterminer le nombre de tâches à continuer d'exécuter dans le service. Par exemple, si vous disposez d'un service qui contient `tasksetA` et que vous créez un `tasksetB`, vous pouvez tester la validité de `tasksetB` avant de vouloir lui transférer du

trafic de production. Vous pouvez régler l'`scale` pour les deux ensembles de tâches sur `100`, et une fois prêt à transférer tout le trafic de production sur `tasksetB`, vous pouvez mettre à jour l'`scale` de `tasksetA` sur `0` pour le diminuer.

CodeDeploy déploiements bleu/vert pour Amazon ECS

Nous vous recommandons d'utiliser le blue/green déploiement Amazon ECS. Pour de plus amples informations, veuillez consulter Création d'un blue/green déploiement Amazon ECS .

Le type de déploiement bleu/vert utilise le modèle de blue/green déploiement contrôlé par. CodeDeploy Ce type de déploiement permet de vérifier un nouveau déploiement d'un service avant d'y envoyer du trafic de production. Pour plus d'informations, consultez le contenu du guide CodeDeploy de l'AWS CodeDeploy utilisateur. Valider l'état d'un service Amazon ECS avant le déploiement

Le trafic peut être modifié de trois manières au cours d'un blue/green déploiement :

- Canary — Le trafic est décalé en deux étapes. Vous pouvez choisir parmi les options canary prédéfinies qui définissent le pourcentage de trafic déplacé vers l'ensemble de tâches mises à jour dans le premier incrément, et l'intervalle (en minutes) avant que le trafic restant soit déplacé dans le second incrément.
- Linéaire — Le trafic est décalé par incréments égaux, avec un nombre égal de minutes entre chaque incrément. Vous pouvez choisir parmi les options linéaires prédéfinies qui définissent le pourcentage de trafic déplacé pour chaque incrément et le nombre de minutes entre chaque incrément.
- R ll-at-once — Tout le trafic est transféré de l'ensemble de tâches d'origine à l'ensemble de tâches mis à jour en une seule fois.

Les composants suivants sont utilisés par Amazon ECS lorsqu'un service utilise le type de blue/green déploiement : CodeDeploy

CodeDeploy application

Une collection de CodeDeploy ressources. Comprend un ou plusieurs groupes de déploiement.

CodeDeploy groupe de déploiement

Paramètres de déploiement. Regroupe les éléments suivants :

- Cluster et service Amazon ECS

- Groupe cible de l'équilibreur de charge et informations sur le port d'écoute
- Stratégie de restauration du déploiement
- Paramètres de réacheminement du trafic
- Paramètres d'arrêt de la révision initiale
- Configuration de déploiement
- CloudWatch configuration des alarmes pouvant être configurée pour arrêter les déploiements
- Paramètres SNS ou CloudWatch Events pour les notifications

Pour de plus amples informations, veuillez consulter Utilisation des groupes de déploiement dans CodeDeploy dans le Guide de l'utilisateur AWS CodeDeploy .

CodeDeploy configuration de déploiement

Spécifie comment CodeDeploy achemine le trafic de production vers votre ensemble de tâches de remplacement lors d'un déploiement. Les configurations de déploiement linéaire et canary prédéfinies suivantes sont disponibles. Vous pouvez également créer des déploiements canary et linéaires personnalisés. Pour obtenir de plus amples informations, consultez Utilisation des configurations de déploiement dans le Guide de l'utilisateur AWS CodeDeploy .

- CodeDeployDefault. ECSAllAtOnce: Transfère immédiatement tout le trafic vers le conteneur Amazon ECS mis à jour
- CodeDeployDefault. ECSLinear10 PercentEvery 1Minutes : déplace 10 % du trafic toutes les minutes jusqu'à ce que tout le trafic soit transféré.
- CodeDeployDefault. ECSLinear10 à PercentEvery 3 minutes : déplace 10 % du trafic toutes les 3 minutes jusqu'à ce que tout le trafic soit transféré.
- CodeDeployDefault. ECSCanary10 % 5 minutes : déplace 10 % du trafic dès le premier incrément. Les 90 % restants sont déployés 5 minutes plus tard.
- CodeDeployDefault. ECSCanary10 % 15 minutes : déplace 10 % du trafic dès le premier incrément. Les 90 % restants sont déployés 15 minutes plus tard.

Révision

Une révision est le fichier de spécification de CodeDeploy l'application (AppSpec fichier). Dans le AppSpec fichier, vous spécifiez l'ARN complet de la définition de tâche ainsi que le conteneur et le port de votre ensemble de tâches de remplacement où le trafic doit être acheminé lors de la création d'un nouveau déploiement. Le nom du conteneur doit être un des noms de conteneur référencés dans votre définition de tâche. Si la configuration réseau ou la version de plate-forme a été mise à jour dans la définition du service, vous devez également spécifier

ces détails dans le AppSpec fichier. Vous pouvez également spécifier les fonctions Lambda à exécuter au cours des événements de cycle de vie du déploiement. Les fonctions Lambda vous permettent d'exécuter des tests et de renvoyer des métriques pendant le déploiement. Pour plus d'informations, consultez la section Référence des AppSpec fichiers dans le guide de AWS CodeDeploy l'utilisateur.

Considérations

Tenez compte des points suivants lorsque vous utilisez le type de blue/green déploiement :

- Lorsqu'un service Amazon ECS utilisant le type de blue/green déploiement est initialement créé, un ensemble de tâches Amazon ECS est créé.
- Vous devez configurer le service pour utiliser un équilibreur de charge Application Load Balancer ou un Network Load Balancer. Les exigences de l'équilibreur de charge sont les suivantes :
  - Vous devez ajouter un port d'écoute de production à l'équilibreur de charge, utilisé pour acheminer le trafic de production.
  - Vous pouvez également ajouter à l'équilibreur de charge un port d'écoute de test, utilisé pour acheminer le trafic de test. Si vous spécifiez un écouteur de test, votre trafic de test est CodeDeploy acheminé vers l'ensemble de tâches de remplacement lors d'un déploiement.
  - Les ports d'écoute de production et de test doivent appartenir au même équilibreur de charge.
  - Vous devez définir un groupe cible pour l'équilibreur de charge. Le groupe cible achemine le trafic vers la tâche initiale dans un service via le port d'écoute de production.
  - Lorsqu'un Network Load Balancer est utilisé, seule la configuration de déploiement `CodeDeployDefault.ECSAllAtOnce` est prise en charge.
- Pour les services configurés pour utiliser Service Auto Scaling et le type de déploiement bleu/vert, la scalabilité automatique n'est pas bloquée pendant un déploiement, mais le déploiement peut échouer dans certaines circonstances. Ce comportement est décrit plus en détail ci-dessous.
  - Si un service évolue et qu'un déploiement démarre, l'ensemble de tâches vertes est créé et CodeDeploy attendra jusqu'à une heure pour qu'il atteigne un état stable et ne transférera aucun trafic tant que ce n'est pas le cas.
  - Si un service est en cours de blue/green déploiement et qu'un événement de dimensionnement se produit, le trafic continuera de changer pendant 5 minutes. Si le service n'atteint pas son état stable dans les 5 minutes, il CodeDeploy arrêtera le déploiement et le marquera comme un échec.

- Les tâches qui utilisent le type de lancement Fargate ou les types de contrôleur de déploiement `CODE_DEPLOY` ne prennent pas en charge la stratégie de planification `DAEMON`.
- Lorsque vous créez initialement une CodeDeploy application et un groupe de déploiement, vous devez spécifier les éléments suivants :
  - Vous devez définir deux groupes cibles pour l'équilibreur de charge. L'un d'eux doit être le groupe cible initial défini pour l'équilibreur de charge lors de la création du service Amazon ECS service. La seule exigence relative au second groupe cible est qu'il ne peut pas être associé à un autre équilibreur de charge que celui utilisé par le service.
- Lorsque vous créez un CodeDeploy déploiement pour un service Amazon ECS, CodeDeploy un ensemble de tâches de remplacement (ou ensemble de tâches vertes) est créé dans le déploiement. Si vous avez ajouté un écouteur de test à l'équilibreur de charge, votre trafic de test est CodeDeploy acheminé vers l'ensemble de tâches de remplacement. Cela indique le moment où vous pouvez exécuter les tests de validation. Ensuite, CodeDeploy réachemine le trafic de production depuis l'ensemble de tâches initial vers l'ensemble des tâches de remplacement en fonction des paramètres de redirection du trafic définis pour le groupe de déploiement.

Autorisations IAM requises

Blue/green deployments are made possible by a combination of the Amazon ECS and CodeDeploy APIs. Users must have the appropriate permissions for these services before they can use Amazon ECS blue/greendéploiements dans le AWS Management Console ou avec le bloc opératoire AWS CLI . SDKs

En plus des autorisations IAM standard nécessaires à la création et à la mise à jour de services, Amazon ECS exige les autorisations suivantes. Ces autorisations ont été ajoutées à la politique IAM `AmazonECS_FullAccess`. Pour de plus amples informations, veuillez consulter Amazon ECS_FullAccess.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codedeploy:CreateApplication",
                "codedeploy:CreateDeployment",
                "codedeploy:CreateDeploymentGroup",
                "codedeploy:GetApplication",
```

```
                "codedeploy:GetDeployment",
                "codedeploy:GetDeploymentGroup",
                "codedeploy:ListApplications",
                "codedeploy:ListDeploymentGroups",
                "codedeploy:ListDeployments",
                "codedeploy:StopDeployment",
                "codedeploy:GetDeploymentTarget",
                "codedeploy:ListDeploymentTargets",
                "codedeploy:GetDeploymentConfig",
                "codedeploy:GetApplicationRevision",
                "codedeploy:RegisterApplicationRevision",
                "codedeploy:BatchGetApplicationRevisions",
                "codedeploy:BatchGetDeploymentGroups",
                "codedeploy:BatchGetDeployments",
                "codedeploy:BatchGetApplications",
                "codedeploy:ListApplicationRevisions",
                "codedeploy:ListDeploymentConfigs",
                "codedeploy:ContinueDeployment",
                "sns:ListTopics",
                "cloudwatch:DescribeAlarms",
                "lambda:ListFunctions"
            ],
            "Resource": ["*"]
        }
    ]
}
```

> **Note**
>
> Outre les autorisations Amazon ECS standard nécessaires pour exécuter des tâches et des services, les utilisateurs doivent également disposer d'autorisations `iam:PassRole` pour pouvoir utiliser les rôles IAM dans le cadre des tâches.

CodeDeploy nécessite des autorisations pour appeler Amazon ECS APIs, modifier votre Elastic Load Balancing, invoquer des fonctions Lambda et décrire les CloudWatch alarmes, ainsi que des autorisations pour modifier le nombre souhaité pour votre service en votre nom. Avant de créer un service Amazon ECS utilisant le type de blue/green déploiement, vous devez créer un rôle IAM (`ecsCodeDeployRole`). Pour de plus amples informations, veuillez consulter Rôle CodeDeploy IAM d'Amazon ECS.

Les exemples de politique Créer un exemple de service Amazon ECS et Exemple de mise à jour du service Amazon ECS IAM montrent les autorisations requises pour que les utilisateurs puissent utiliser les blue/green déploiements Amazon ECS sur le. AWS Management Console

Migrer les CodeDeploy blue/green deployments to Amazon ECS blue/green déploiements

CodeDeploy blue/green and Amazon ECS blue/greenles déploiements fournissent des fonctionnalités similaires, mais ils diffèrent dans la façon dont vous les configurez et les gérez.

CodeDeploy aperçu du déploiement bleu/vert

Lorsque vous créez un service Amazon ECS à l'aide de CodeDeploy, vous :

1. Créez un équilibreur de charge avec un écouteur de production et (éventuellement) un écouteur de test. Chaque écouteur est configuré avec une seule règle (par défaut) qui achemine tout le trafic vers un seul groupe cible (le groupe cible principal).
2. Créez un service Amazon ECS, configuré pour utiliser l'écouteur et le groupe cible, avec le `deploymentController` type défini sur. `CODE_DEPLOY` La création d'un service entraîne la création d'un ensemble de tâches (bleu) enregistré auprès du groupe cible spécifié.
3. Créez un groupe de CodeDeploy déploiement (dans le cadre d'une CodeDeploy application) et configurez-le avec les détails du cluster Amazon ECS, le nom du service, les écouteurs d'équilibrage de charge, deux groupes cibles (le groupe cible principal utilisé dans la règle d'écoute de production et un groupe cible secondaire à utiliser pour les tâches de remplacement), un rôle de service (pour accorder des CodeDeploy autorisations pour manipuler les ressources Amazon ECS et Elastic Load Balancing) et divers paramètres qui contrôlent le comportement de déploiement.

Avec CodeDeploy, les nouvelles versions d'un service sont déployées à l'aide `CreateDeployment()` du nom de l' CodeDeploy application, du nom du groupe de déploiement et d'un AppSpec fichier fournissant les détails de la nouvelle révision et des hooks de cycle de vie facultatifs. Le CodeDeploy déploiement crée un ensemble de tâches de remplacement (vert) et enregistre ses tâches auprès du groupe cible secondaire. Lorsque cela devient sain, il est disponible pour les tests (facultatif) et pour la production. Dans les deux cas, le réacheminement est réalisé en modifiant la règle d'écoute correspondante pour qu'elle pointe vers le groupe cible secondaire associé à l'ensemble de tâches vert. Le rollback est obtenu en redéfinissant la règle de l'écouteur de production pour la ramener au groupe cible principal.

Présentation du blue/green déploiement d'Amazon ECS

Dans le cas des blue/green déploiements Amazon ECS, la configuration du déploiement fait partie du service Amazon ECS lui-même :

1. Vous devez préconfigurer l'écouteur de production de l'équilibreur de charge avec une règle qui inclut deux groupes cibles avec des pondérations de 1 et 0.
2. Vous devez spécifier les ressources suivantes ou mettre à jour les ressources du service :
   - L'ARN de cette règle d'écoute
   - Les deux groupes cibles
   - Un rôle IAM pour autoriser Amazon ECS à appeler Elastic Load Balancing APIs
   - Rôle IAM facultatif pour exécuter les fonctions Lambda
   - Définissez `deploymentController` le type sur `ECS` et `deploymentConfiguration.strategy` sur`BLUE_GREEN`. Cela entraîne la création d'un déploiement de service (bleu) dont les tâches sont enregistrées auprès du groupe cible principal.

Avec Amazon ECS bleu/vert, une nouvelle révision de service est créée en appelant Amazon ECS et en transmettant `UpdateService()` les détails de la nouvelle révision. Le déploiement du service crée de nouvelles tâches de révision de service (vertes) et les enregistre auprès du groupe cible secondaire. Amazon ECS gère les opérations de réacheminement et de rollback en modifiant les pondérations selon la règle de l'écouteur.

Principales différences de mise en œuvre

Bien que les deux approches aboutissent à la création d'un ensemble initial de tâches, la mise en œuvre sous-jacente est différente :

- CodeDeploy utilise un ensemble de tâches, tandis qu'Amazon ECS utilise une révision de service. Les ensembles de tâches Amazon ECS sont une ancienne structure qui a été remplacée par les révisions et les déploiements des services Amazon ECS. Ces derniers offrent une meilleure visibilité sur le processus de déploiement, ainsi que sur l'historique des déploiements et des révisions des services.
- Avec CodeDeploy, les hooks du cycle de vie sont spécifiés dans le cadre du AppSpec fichier fourni à`CreateDeployment()`. Cela signifie que les crochets peuvent être modifiés d'un déploiement à l'autre. Avec Amazon ECS bleu/vert, les hooks sont spécifiés dans le cadre de la configuration du service, et toute mise à jour nécessite un `UpdateService()` appel.

- Amazon ECS CodeDeploy et Amazon ECS blue/green utilisent Lambda pour implémenter le hook, mais les entrées et sorties attendues sont différentes.

  Avec CodeDeploy, la fonction doit appeler `PutLifecycleEventHookExecutionStatus()` pour renvoyer le statut du hook, qui peut être `SUCCEEDED` ou`FAILED`. Avec Amazon ECS, la réponse Lambda est utilisée pour indiquer l'état du hook.
- CodeDeploy invoque chaque hook en tant qu'appel ponctuel et s'attend à ce qu'un statut d'exécution final soit renvoyé dans un délai d'une heure. Les hooks Amazon ECS sont plus flexibles dans la mesure où ils peuvent renvoyer un `IN_PROGRESS` indicateur, qui indique que le hook doit être réinvoqué à plusieurs reprises jusqu'à ce qu'il aboutisse à `SUCCEEDED` ou`FAILED`. Pour de plus amples informations, veuillez consulter Hooks relatifs au cycle de vie pour les déploiements de services Amazon ECS.

Approches de migration

Il existe trois approches principales pour migrer à partir de CodeDeploy blue/green to Amazon ECS blue/green déploiements. Chaque approche présente des caractéristiques différentes en termes de complexité, de risque, de capacité de réduction et de temps d'arrêt potentiel.

Réutilisez les mêmes ressources Elastic Load Balancing que celles utilisées pour CodeDeploy

Vous mettez à jour le service Amazon ECS existant pour utiliser le contrôleur de déploiement Amazon ECS avec une stratégie de blue/green déploiement au lieu du contrôleur de CodeDeploy déploiement. Lorsque vous utilisez cette approche, tenez compte des points suivants :

- La procédure de migration est plus simple car vous mettez à jour le contrôleur de déploiement du service Amazon ECS et la stratégie de déploiement existants.
- Il n'y a aucun temps d'arrêt lorsque la configuration et la migration sont correctes.
- Pour effectuer une annulation, vous devez annuler la révision du service.
- Le risque est élevé car il n'existe pas de configuration parallèle bleu/vert.

Vous utilisez le même écouteur d'équilibrage de charge et les mêmes groupes cibles que ceux utilisés pour. CodeDeploy Si vous utilisez AWS CloudFormation, consultezMigration d'un modèle AWS CloudFormation CodeDeploy blue/green deployment template to an Amazon ECS blue/green AWS CloudFormation.

1. Modifiez la règle par défaut des production/test auditeurs pour inclure le groupe cible alternatif et définissez le poids du groupe cible principal sur 1 et du groupe cible alternatif sur 0.

   En CodeDeploy effet, les écouteurs de l'équilibreur de charge rattachés au service sont configurés avec une règle unique (par défaut) qui achemine tout le trafic vers un seul groupe cible. Pour le bleu/vert d'Amazon ECS, les écouteurs de l'équilibreur de charge doivent être préconfigurés avec une règle qui inclut les deux groupes cibles avec des pondérations.
2. Mettez à jour le service Amazon ECS existant en appelant l'`UpdateService`API et en définissant le paramètre `deploymentController` sur et le paramètre `deploymentStrategy` sur`BLUE_GREEN`. `ECS` Spécifiez le ARNs groupe cible, le groupe cible alternatif, l'écouteur de production et un écouteur de test facultatif.
3. Vérifiez que le service fonctionne comme prévu.
4. Supprimez la CodeDeploy configuration de ce service Amazon ECS car vous utilisez désormais Amazon ECS bleu/vert.

Nouveau service avec équilibreur de charge existant

Cette approche utilise la blue/green stratégie de migration.

Lorsque vous utilisez cette approche, tenez compte des points suivants :

- Les perturbations sont minimes. Cela se produit uniquement lors du changement de port d'Elastic Load Balancing.
- Pour effectuer un rollback, vous devez annuler le changement de port d'Elastic Load Balancing.
- Le risque est faible car il existe des configurations parallèles. Vous pouvez donc effectuer un test avant le changement de trafic.

1. Laissez les écouteurs, les groupes cibles et le service Amazon ECS intacts pour la CodeDeploy configuration afin de pouvoir facilement revenir à cette configuration si nécessaire.
2. Créez de nouveaux groupes cibles et de nouveaux écouteurs (avec des ports différents de ceux des écouteurs d'origine) sous l'équilibreur de charge existant. Créez ensuite un nouveau service Amazon ECS qui correspond au service Amazon ECS existant, sauf que vous l'utilisez `ECS` comme contrôleur de déploiement, `BLUE_GREEN` comme stratégie de déploiement, et transmettez les règles ARNs pour les nouveaux groupes cibles et les nouveaux auditeurs.

3. Vérifiez la nouvelle configuration en envoyant manuellement le trafic HTTP au service. Si tout se passe bien, échangez les ports des écouteurs d'origine et des nouveaux écouteurs pour acheminer le trafic vers la nouvelle configuration.
4. Vérifiez la nouvelle configuration et, si tout continue de fonctionner comme prévu, CodeDeploy supprimez-la.

Nouveau service avec un nouvel équilibreur de charge

Comme l'approche précédente, cette approche utilise la blue/green stratégie de migration. La principale différence est que le passage de la CodeDeploy configuration à la configuration Amazon ECS s'effectue au niveau blue/green d'une couche proxy inverse située au-dessus de l'équilibreur de charge. Des exemples d'implémentations pour la couche proxy inverse sont Route 53 et CloudFront.

Cette approche convient aux clients qui disposent déjà de cette couche proxy inverse, et si toutes les communications avec le service passent par cette couche (par exemple, aucune communication directe au niveau de l'équilibreur de charge).

Lorsque vous utilisez cette approche, tenez compte des points suivants :

- Cela nécessite une couche proxy inverse.
- La procédure de migration est plus complexe car vous devez mettre à jour le contrôleur de déploiement du service Amazon ECS et la stratégie de déploiement existants.
- Les perturbations sont minimes. Cela se produit uniquement lors du changement de port d'Elastic Load Balancing.
- Pour effectuer une restauration, vous devez annuler les modifications apportées à la configuration du proxy.
- Le risque est faible car il existe des configurations parallèles. Vous pouvez donc effectuer un test avant le changement de trafic.

1. Ne modifiez pas la CodeDeploy configuration existante intacte (équilibreur de charge, écouteurs, groupes cibles, service Amazon ECS et groupe de CodeDeploy déploiement).
2. Créez un nouvel équilibreur de charge, des groupes cibles et des écouteurs configurés pour les déploiements Amazon ECS. blue/green

   Configurez les ressources appropriées.

- Application Load Balancer - Pour plus d'informations, consultez. Ressources d'Application Load Balancer pour les déploiements blue/green
- Network Load Balancer - Pour plus d'informations, consultez. Ressources Network Load Balancer pour les déploiements bleu/vert d'Amazon ECS

3. Créez un nouveau service en `ECS` tant que contrôleur de déploiement et en `BLUE_GREEN` tant que stratégie de déploiement, en pointant vers les nouvelles ressources de l'équilibreur de charge.
4. Vérifiez la nouvelle configuration en la testant via le nouvel équilibreur de charge.
5. Mettez à jour la configuration du proxy inverse pour acheminer le trafic vers le nouvel équilibreur de charge.
6. Observez la nouvelle révision du service et, si tout continue de fonctionner comme prévu, supprimez la CodeDeploy configuration.

Étapes suivantes

Après la migration vers les blue/green déploiements Amazon ECS :

- Mettez à jour vos scripts et CI/CD pipelines de déploiement pour utiliser l'`UpdateService`API Amazon ECS au lieu de l' CodeDeploy `CreateDeployment`API.
- Mettez à jour votre surveillance et vos alertes pour suivre les déploiements de services Amazon ECS plutôt que les déploiements. CodeDeploy
- Envisagez de mettre en œuvre des tests automatisés de votre nouveau processus de déploiement pour vous assurer qu'il fonctionne comme prévu.

Migration depuis un déploiement de CodeDeploy blue/green to an Amazon ECS blue/green service

En utilisant les blue/green déploiements Amazon ECS, vous pouvez apporter et tester des modifications de service avant de les implémenter dans un environnement de production.

Vous devez créer de nouveaux hooks de cycle de vie pour votre blue/green déploiement Amazon ECS.

Prérequis

Effectuez les opérations suivantes avant de démarrer un blue/green déploiement.

1. Remplacez le rôle Amazon ECS CodeDeploy IAM par les autorisations suivantes.

- Pour plus d'informations sur les autorisations d'Elastic Load Balancing, consultezRôle IAM de l'infrastructure Amazon ECS pour les équilibreurs de charge.
- Pour plus d'informations sur les autorisations Lambda, consultez. Autorisations requises pour les fonctions Lambda dans les déploiements Amazon ECS blue/green

2. Désactivez CodeDeploy l'automatisation. Pour plus d'informations, consultez la section Utilisation des groupes de déploiement CodeDeploy dans le Guide de CodeDeploy l'utilisateur.
3. Vérifiez que vous disposez des informations suivantes issues de votre CodeDeploy blue/green deployment. You can reuse this information for the Amazon ECS blue/green déploiement :
   - Le groupe cible de production
   - L'auditeur de production
   - La règle de production
   - Le groupe cible du test

     C'est le groupe cible de la révision du service écologique,
4. Assurez-vous que les groupes cibles de votre Application Load Balancer sont correctement associés aux règles d'écoute :
   - Si vous n'utilisez pas d'écouteurs de test, les deux groupes cibles (production et test) doivent être associés aux règles d'écoute de production.
   - Si vous utilisez des écouteurs de test, un groupe cible doit être lié aux règles d'écoute de production et l'autre groupe cible doit être lié aux règles d'écoute de test.

   Si cette exigence n'est pas remplie, le déploiement du service échouera avec l'erreur suivante : `Service deployment rolled back because of invalid networking configuration. Both targetGroup and alternateTargetGroup must be associated with the productionListenerRule or testListenerRule.`
5. Vérifiez qu'aucun déploiement de service n'est en cours pour le service. Pour de plus amples informations, veuillez consulter Afficher l'historique des services à l'aide des déploiements de services Amazon ECS.
6. Les blue/green déploiements Amazon ECS nécessitent que votre service utilise l'une des fonctionnalités suivantes : Configurez les ressources appropriées.
   - Application Load Balancer - Pour plus d'informations, consultez. Ressources d'Application Load Balancer pour les déploiements blue/green
   - Network Load Balancer - Pour plus d'informations, consultez. Ressources Network Load Balancer pour les déploiements bleu/vert d'Amazon ECS

- Service Connect - Pour plus d'informations, consultezRessources Service Connect pour les déploiements bleu/vert d'Amazon ECS.

7. Décidez si vous souhaitez exécuter les fonctions Lambda pendant les étapes du cycle de vie des étapes du déploiement d'Amazon ECS blue/green .
   - Avant la mise à l'échelle
   - Après la mise à l'échelle
   - Testez le transfert de trafic
   - Après le test, changement de trafic
   - Réorientation du trafic de production
   - Après le transfert du trafic de production

   Créez des fonctions Lambda pour chaque étape du cycle de vie. Pour plus d'informations, voir Création d'une fonction Lambda avec la console dans le Guide du AWS Lambda développeur.

Pour plus d'informations sur la mise à jour du contrôleur de déploiement d'un service, consultezMettre à jour les paramètres du service Amazon ECS.

Procédure

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.

   La page de détails du cluster s'affiche.
3. Dans l'onglet Services, sélectionnez le service.

   La page des détails du service s'affiche.
4. Dans la bannière, choisissez Mettre à jour le type de contrôleur de déploiement.

   La page du type de contrôleur de déploiement Migrate s'affiche.
5. Développez Nouveau, puis spécifiez les paramètres suivants.
   a. Pour le type de contrôleur de déploiement, choisissez ECS.
   b. Pour la stratégie de déploiement, choisissez Bleu/vert.
   c. Pour le temps de cuisson, entrez le temps pendant lequel les révisions de service bleues et vertes s'exécutent.

d. Pour exécuter des fonctions Lambda pendant une étape du cycle de vie, sous Deployment lifecyce hooks, effectuez les opérations suivantes pour chaque fonction Lambda unique :

   i. Choisissez Ajouter.

      Répétez l'opération pour chaque fonction unique que vous souhaitez exécuter.

   ii. Pour la fonction Lambda, entrez le nom de la fonction.

   iii. Pour Rôle, choisissez le rôle que vous avez créé dans les prérequis avec les autorisations bleu/vert.

      Pour de plus amples informations, veuillez consulter Autorisations requises pour les fonctions Lambda dans les déploiements Amazon ECS blue/green .

   iv. Pour les étapes du cycle de vie, sélectionnez les étapes exécutées par la fonction Lambda.

   v. (Facultatif) Pour les détails du crochet, entrez une paire clé-valeur fournissant des informations sur le crochet.

6. Développez l'équilibrage de charge, puis configurez les éléments suivants :

   a. Pour Rôle, choisissez le rôle que vous avez créé dans les prérequis avec les blue/green autorisations.

      Pour de plus amples informations, veuillez consulter Autorisations requises pour les fonctions Lambda dans les déploiements Amazon ECS blue/green .

   b. Pour Listener, choisissez l'écouteur de production dans votre déploiement CodeDeploy bleu/vert.

   c. Pour Règle de production, choisissez la règle de production dans votre déploiement CodeDeploy bleu/vert.

   d. Pour Règle de test, choisissez la règle de test de votre déploiement CodeDeploy bleu/vert.

   e. Pour Groupe cible, choisissez le groupe cible de production dans votre déploiement CodeDeploy bleu/vert.

   f. Pour Groupe cible alternatif, choisissez le groupe cible de test dans votre déploiement CodeDeploy bleu/vert.

7. Choisissez Mettre à jour.

Étapes suivantes

- Mettez à jour le service pour démarrer le déploiement. Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.
- Surveillez le processus de déploiement pour vous assurer qu'il suit le schéma bleu/vert :
  - La révision du service écologique est créée et étendue
  - Le trafic de test est acheminé vers la version verte (si elle est configurée)
  - Le trafic de production passe à la version verte
  - Après le temps de cuisson, la révision bleue est terminée

Migration d'un modèle AWS CloudFormation CodeDeploy blue/green deployment template to an Amazon ECS blue/green AWS CloudFormation

Migrez un AWS CloudFormation modèle qui utilise une stratégie CodeDeploy blue/green deployments for Amazon ECS services to one that uses the native Amazon ECS blue/green de déploiement. La migration suit l'approche « Réutilisez les mêmes ressources Elastic Load Balancing utilisées pour CodeDeploy ». Pour plus d'informations, consultez the section called “Migrer les CodeDeploy blue/green deployments to Amazon ECS blue/green déploiements” la documentation.

Modèle de source

Ce modèle utilise la `AWS::CodeDeployBlueGreen` transformation et le `AWS::CodeDeploy::BlueGreen` hook pour implémenter blue/green des déploiements pour un service Amazon ECS.

Source

Il s'agit du AWS CloudFormation modèle complet utilisant le déploiement CodeDeploy bleu/vert. Pour plus d'informations, consultez l'exemple de modèle de déploiement bleu/vert dans le guide de l' AWS CloudFormation utilisateur :

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "Vpc": {
      "Type": "AWS::EC2::VPC::Id"
    },
    "Subnet1": {
      "Type": "AWS::EC2::Subnet::Id"
```

```
    },
    "Subnet2": {
      "Type": "AWS::EC2::Subnet::Id"
    }
  },
  "Transform": [
    "AWS::CodeDeployBlueGreen"
  ],
  "Hooks": {
    "CodeDeployBlueGreenHook": {
      "Type": "AWS::CodeDeploy::BlueGreen",
      "Properties": {
        "TrafficRoutingConfig": {
          "Type": "TimeBasedCanary",
          "TimeBasedCanary": {
            "StepPercentage": 15,
            "BakeTimeMins": 5
          }
        },
        "Applications": [
          {
            "Target": {
              "Type": "AWS::ECS::Service",
              "LogicalID": "ECSDemoService"
            },
            "ECSAttributes": {
              "TaskDefinitions": [
                "BlueTaskDefinition",
                "GreenTaskDefinition"
              ],
              "TaskSets": [
                "BlueTaskSet",
                "GreenTaskSet"
              ],
              "TrafficRouting": {
                "ProdTrafficRoute": {
                  "Type": "AWS::ElasticLoadBalancingV2::Listener",
                  "LogicalID": "ALBListenerProdTraffic"
                },
                "TargetGroups": [
                  "ALBTargetGroupBlue",
                  "ALBTargetGroupGreen"
                ]
              }
```

```
            }
          }
        ]
      }
    }
  },
  "Resources": {
    "ExampleSecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "GroupDescription": "Security group for ec2 access",
        "VpcId": {"Ref": "Vpc"},
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 8080,
            "ToPort": 8080,
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIp": "0.0.0.0/0"
          }
        ]
      }
    },
    "ALBTargetGroupBlue": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
```

```
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ALBTargetGroupGreen": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ExampleALB": {
      "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
      "Properties": {
        "Scheme": "internet-facing",
```

```
        "SecurityGroups": [
          {"Ref": "ExampleSecurityGroup"}
        ],
        "Subnets": [
          {"Ref": "Subnet1"},
          {"Ref": "Subnet2"}
        ],
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "Type": "application",
        "IpAddressType": "ipv4"
      }
    },
    "ALBListenerProdTraffic": {
      "Type": "AWS::ElasticLoadBalancingV2::Listener",
      "Properties": {
        "DefaultActions": [
          {
            "Type": "forward",
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                }
              ]
            }
          }
        ],
        "LoadBalancerArn": {"Ref": "ExampleALB"},
        "Port": 80,
        "Protocol": "HTTP"
      }
    },
    "ALBListenerProdRule": {
      "Type": "AWS::ElasticLoadBalancingV2::ListenerRule",
      "Properties": {
        "Actions": [
          {
            "Type": "forward",
```

```
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                }
              ]
            }
          }
        ],
        "Conditions": [
          {
            "Field": "http-header",
            "HttpHeaderConfig": {
              "HttpHeaderName": "User-Agent",
              "Values": [
                "Mozilla"
              ]
            }
          }
        ],
        "ListenerArn": {"Ref": "ALBListenerProdTraffic"},
        "Priority": 1
      }
    },
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "",
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
```

```
      }
    },
    "BlueTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
        "ContainerDefinitions": [
          {
            "Name": "DemoApp",
            "Image": "nginxdemos/hello:latest",
            "Essential": true,
            "PortMappings": [
              {
                "HostPort": 80,
                "Protocol": "tcp",
                "ContainerPort": 80
              }
            ]
          }
        ],
        "RequiresCompatibilities": [
          "FARGATE"
        ],
        "NetworkMode": "awsvpc",
        "Cpu": "256",
        "Memory": "512",
        "Family": "ecs-demo"
      }
    },
    "ECSDemoCluster": {
      "Type": "AWS::ECS::Cluster",
      "Properties": {}
    },
    "ECSDemoService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "DesiredCount": 1,
        "DeploymentController": {
          "Type": "EXTERNAL"
        }
      }
    },
    "BlueTaskSet": {
```

```
      "Type": "AWS::ECS::TaskSet",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "LaunchType": "FARGATE",
        "NetworkConfiguration": {
          "AwsVpcConfiguration": {
            "AssignPublicIp": "ENABLED",
            "SecurityGroups": [
              {"Ref": "ExampleSecurityGroup"}
            ],
            "Subnets": [
              {"Ref": "Subnet1"},
              {"Ref": "Subnet2"}
            ]
          }
        },
        "PlatformVersion": "1.4.0",
        "Scale": {
          "Unit": "PERCENT",
          "Value": 100
        },
        "Service": {"Ref": "ECSDemoService"},
        "TaskDefinition": {"Ref": "BlueTaskDefinition"},
        "LoadBalancers": [
          {
            "ContainerName": "DemoApp",
            "ContainerPort": 80,
            "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"}
          }
        ]
      }
    },
    "PrimaryTaskSet": {
      "Type": "AWS::ECS::PrimaryTaskSet",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "Service": {"Ref": "ECSDemoService"},
        "TaskSetId": {"Fn::GetAtt": ["BlueTaskSet", "Id"]}
      }
    }
  }
}
```

Étapes de la migration

Supprimer des CodeDeploy ressources spécifiques

Vous n'avez plus besoin des propriétés suivantes :

- La `AWS::CodeDeployBlueGreen` transformation
- Le `CodeDeployBlueGreenHook` crochet
- Les `GreenTaskSet` ressources `GreenTaskDefinition` et (elles seront gérées par Amazon ECS)
- La `PrimaryTaskSet` ressource (Amazon ECS gérera les ensembles de tâches en interne)

Reconfigurer l'écouteur de l'équilibreur de charge

Modifiez la `ALBListenerProdTraffic` ressource pour utiliser une action directe avec deux groupes cibles :

```
{
  "DefaultActions": [
    {
      "Type": "forward",
      "ForwardConfig": {
        "TargetGroups": [
          {
            "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
            "Weight": 1
          },
          {
            "TargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
            "Weight": 0
          }
        ]
      }
    }
  ]
}
```

Mettre à jour les propriétés de déploiement

Mettez à jour et ajoutez les éléments suivants :

- Modifiez la `DeploymentController` propriété de `EXTERNAL` à`ECS`.
- Ajoutez la `Strategy` propriété et définissez-la sur BLUE_GREEN.
- Ajoutez la `BakeTimeInMinutes` propriété.

```
{
  "DeploymentConfiguration": {
    "MaximumPercent": 200,
    "MinimumHealthyPercent": 100,
    "DeploymentCircuitBreaker": {
      "Enable": true,
      "Rollback": true
    },
    "BakeTimeInMinutes": 5,
    "Strategy": "BLUE_GREEN"
  }
}
```

- Ajoutez la configuration de l'équilibreur de charge au service :

```
{
  "LoadBalancers": [
    {
      "ContainerName": "DemoApp",
      "ContainerPort": 80,
      "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
      "AdvancedConfiguration": {
        "AlternateTargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
        "ProductionListenerRule": {"Ref": "ALBListenerProdRule"},
        "RoleArn": {"Fn::GetAtt": ["ECSInfrastructureRoleForLoadBalancers", "Arn"]}
      }
    }
  ]
}
```

- Ajoutez la référence de définition de tâche au service :

```
{
  "TaskDefinition": {"Ref": "BlueTaskDefinition"}
}
```

Création du ECSInfrastructure RolePolicyForLoadBalancers rôle Amazon

Ajoutez un nouveau rôle IAM qui permet à Amazon ECS de gérer les ressources de l'équilibreur de charge :

Remplacez le *user-input* par votre valeur.

```
{
  "ECSInfrastructureRoleForLoadBalancers": {
    "Type": "AWS::IAM::Role",
    "Properties": {
      "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Sid": "AllowAccessToECSForInfrastructureManagement",
            "Effect": "Allow",
            "Principal": {
              "Service": "ecs.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
          }
        ]
      },
      "ManagedPolicyArns": [
        "arn:aws:iam::aws:policy/AmazonECSInfrastructureRolePolicyForLoadBalancers"
      ]
    }
  }
}
```

Recommandations relatives aux tests

1. Déployez le modèle migré dans un environnement hors production.
2. Vérifiez que le service se déploie correctement avec la configuration initiale.
3. Testez un déploiement en mettant à jour la définition de la tâche et en observant le processus de déploiement bleu/vert.
4. Vérifiez que le trafic est correctement transféré entre les déploiements bleu et vert.
5. Testez la fonctionnalité de restauration en forçant un échec de déploiement.

Modèle après migration

Modèle final

Voici le AWS CloudFormation modèle complet utilisant un blue/green déploiement Amazon ECS :

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "Vpc": {
      "Type": "AWS::EC2::VPC::Id"
    },
    "Subnet1": {
      "Type": "AWS::EC2::Subnet::Id"
    },
    "Subnet2": {
      "Type": "AWS::EC2::Subnet::Id"
    }
  },
  "Resources": {
    "ExampleSecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "GroupDescription": "Security group for ec2 access",
        "VpcId": {"Ref": "Vpc"},
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 8080,
            "ToPort": 8080,
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIp": "0.0.0.0/0"
          }
```

```
        ]
      }
    },
    "ALBTargetGroupBlue": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ALBTargetGroupGreen": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
```

```
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ExampleALB": {
      "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
      "Properties": {
        "Scheme": "internet-facing",
        "SecurityGroups": [
          {"Ref": "ExampleSecurityGroup"}
        ],
        "Subnets": [
          {"Ref": "Subnet1"},
          {"Ref": "Subnet2"}
        ],
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "Type": "application",
        "IpAddressType": "ipv4"
      }
    },
    "ALBListenerProdTraffic": {
      "Type": "AWS::ElasticLoadBalancingV2::Listener",
      "Properties": {
        "DefaultActions": [
          {
            "Type": "forward",
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                },
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
```

```
                "Weight": 0
              }
            ]
          }
        }
      ],
      "LoadBalancerArn": {"Ref": "ExampleALB"},
      "Port": 80,
      "Protocol": "HTTP"
    }
  },
  "ALBListenerProdRule": {
    "Type": "AWS::ElasticLoadBalancingV2::ListenerRule",
    "Properties": {
      "Actions": [
        {
          "Type": "forward",
          "ForwardConfig": {
            "TargetGroups": [
              {
                "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                "Weight": 1
              },
              {
                "TargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
                "Weight": 0
              }
            ]
          }
        }
      ],
      "Conditions": [
        {
          "Field": "http-header",
          "HttpHeaderConfig": {
            "HttpHeaderName": "User-Agent",
            "Values": [
              "Mozilla"
            ]
          }
        }
      ],
      "ListenerArn": {"Ref": "ALBListenerProdTraffic"},
      "Priority": 1
```

```
      }
    },
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "",
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
      }
    },
    "ECSInfrastructureRoleForLoadBalancers": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "AllowAccessToECSForInfrastructureManagement",
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/AmazonECSInfrastructureRolePolicyForLoadBalancers"
        ]
      }
    },
```

```
    "BlueTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
        "ContainerDefinitions": [
          {
            "Name": "DemoApp",
            "Image": "nginxdemos/hello:latest",
            "Essential": true,
            "PortMappings": [
              {
                "HostPort": 80,
                "Protocol": "tcp",
                "ContainerPort": 80
              }
            ]
          }
        ],
        "RequiresCompatibilities": [
          "FARGATE"
        ],
        "NetworkMode": "awsvpc",
        "Cpu": "256",
        "Memory": "512",
        "Family": "ecs-demo"
      }
    },
    "ECSDemoCluster": {
      "Type": "AWS::ECS::Cluster",
      "Properties": {}
    },
    "ECSDemoService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "DesiredCount": 1,
        "DeploymentController": {
          "Type": "ECS"
        },
        "DeploymentConfiguration": {
          "MaximumPercent": 200,
          "MinimumHealthyPercent": 100,
          "DeploymentCircuitBreaker": {
            "Enable": true,
```

```
            "Rollback": true
          },
          "BakeTimeInMinutes": 5,
          "Strategy": "BLUE_GREEN"
        },
        "NetworkConfiguration": {
          "AwsvpcConfiguration": {
            "AssignPublicIp": "ENABLED",
            "SecurityGroups": [
              {"Ref": "ExampleSecurityGroup"}
            ],
            "Subnets": [
              {"Ref": "Subnet1"},
              {"Ref": "Subnet2"}
            ]
          }
        },
        "LaunchType": "FARGATE",
        "PlatformVersion": "1.4.0",
        "TaskDefinition": {"Ref": "BlueTaskDefinition"},
        "LoadBalancers": [
          {
            "ContainerName": "DemoApp",
            "ContainerPort": 80,
            "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
            "AdvancedConfiguration": {
              "AlternateTargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
              "ProductionListenerRule": {"Ref": "ALBListenerProdRule"},
              "RoleArn": {"Fn::GetAtt": ["ECSInfrastructureRoleForLoadBalancers",
 "Arn"]}
            }
          }
        ]
      }
    }
  }
}
```

Migration d'un service de mise à jour continue CodeDeploy bleu/vert vers un déploiement de service de mise à jour continue Amazon ECS

Vous pouvez migrer vos déploiements de services d'un déploiement CodeDeploy bleu/vert vers un déploiement de mise à jour continue Amazon ECS. Cela vous permet de passer de la CodeDeploy dépendance à l'utilisation d'un déploiement intégré.

Le planificateur de services Amazon ECS remplace les tâches en cours d'exécution par de nouvelles tâches. Le nombre de tâches ajoutées au service ou supprimées du service par Amazon ECS lors d'une mise à jour propagée est contrôlé par la configuration de déploiement du service.

Prérequis

Effectuez les opérations suivantes avant de démarrer un blue/green déploiement.

1. Vous n'avez plus besoin du rôle CodeDeploy IAM Amazon ECS.
2. Désactivez CodeDeploy l'automatisation. Pour plus d'informations, consultez la section Utilisation des groupes de déploiement CodeDeploy dans le Guide de CodeDeploy l'utilisateur.
3. Vérifiez qu'aucun déploiement de service n'est en cours pour le service. Pour de plus amples informations, veuillez consulter Afficher l'historique des services à l'aide des déploiements de services Amazon ECS.

Pour plus d'informations sur la mise à jour du contrôleur de déploiement d'un service, consultezMettre à jour les paramètres du service Amazon ECS.

Procédure

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.

   La page de détails du cluster s'affiche.
3. Dans l'onglet Services, sélectionnez le service.

   La page des détails du service s'affiche.
4. Dans la bannière, choisissez Migrer.

   La page de configuration du déploiement de la mise à jour s'affiche.
5. Développez les options de déploiement, puis spécifiez les paramètres suivants.

a. Pour le type de contrôleur de déploiement, choisissez ECS.

b. Pour Stratégie de déploiement, choisissez Rolling update.

c. Pour Min running tasks (Minimum de tâches en cours d'exécution), saisissez la limite inférieure pour le nombre de tâches du service qui doivent rester à l'état `RUNNING` lors d'un déploiement, en tant que pourcentage de nombre souhaité de tâches (arrondi au nombre entier supérieur le plus proche). Pour plus d'informations, consultez Deployment configuration (Configuration de déploiement).

d. Pour Max running tasks (Maximum de tâches en cours d'exécution), saisissez la limite supérieure pour le nombre de tâches du service qui peuvent rester à l'état `RUNNING` ou `PENDING` lors d'un déploiement, en tant que pourcentage de nombre souhaité de tâches (arrondi au nombre entier inférieur le plus proche).

6. Développez l'équilibrage de charge, puis configurez les éléments suivants :

   a. Pour Rôle, choisissez le rôle que vous avez créé dans les prérequis avec les blue/green autorisations.

      Pour de plus amples informations, veuillez consulter Autorisations requises pour les fonctions Lambda dans les déploiements Amazon ECS blue/green .

   b. Pour Listener, choisissez l'écouteur de production dans votre déploiement CodeDeploy bleu/vert.

   c. Pour Groupe cible, choisissez le groupe cible de production dans votre déploiement CodeDeploy bleu/vert.

7. Choisissez Mettre à jour.

Étapes suivantes

Vous devez mettre à jour le service pour que la modification prenne effet. Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.

Mettre à jour la stratégie de déploiement d'Amazon ECS

Amazon ECS prend en charge plusieurs stratégies de déploiement pour mettre à jour vos services. Vous pouvez migrer entre ces stratégies en fonction des exigences de votre application. Cette rubrique explique comment migrer entre des déploiements progressifs et des blue/green déploiements.

Comprendre les stratégies de déploiement d'Amazon ECS

Avant de passer d'une stratégie de déploiement à une autre, il est important de comprendre le fonctionnement de chaque stratégie et ses principales différences :

Déploiements progressifs

Dans le cadre d'un déploiement continu, Amazon ECS remplace la version en cours d'exécution de votre application par une nouvelle version. Le planificateur de services utilise les paramètres de pourcentage de santé minimum et maximum pour déterminer la stratégie de déploiement.

Les déploiements progressifs sont plus simples à configurer mais offrent moins de contrôle sur le processus de déploiement et le routage du trafic.

Déploiements bleu/vert

Lors d'un blue/green déploiement, Amazon ECS crée une nouvelle version de votre service (vert) à côté de la version existante (bleu). Cela vous permet de vérifier la nouvelle version avant d'y acheminer le trafic de production.

Les déploiements bleu/vert permettent de mieux contrôler le processus de déploiement, notamment les fonctionnalités de transfert du trafic, de test et de restauration.

Bonnes pratiques

Suivez ces bonnes pratiques lors de la migration entre les stratégies de déploiement :

- Test dans un environnement hors production : testez toujours la mise à jour dans un environnement hors production avant d'appliquer des modifications aux services de production.
- Planification de l'annulation : établissez un plan de restauration au cas où la mise à jour ne fonctionnerait pas comme prévu.
- Surveillance pendant la transition : surveillez de près votre service pendant et après la migration pour vous assurer qu'il continue de fonctionner correctement.
- Mettre à jour la documentation : mettez à jour votre documentation de déploiement pour refléter la nouvelle stratégie de déploiement.
- Tenez compte de l'impact sur le trafic : déterminez l'impact de la mise à jour sur le trafic vers votre service et planifiez en conséquence.

Mise à jour de la stratégie de déploiement depuis la mise à jour progressive vers Amazon ECS bleu/vert

Vous pouvez passer d'un déploiement de mises à jour continues à un blue/green déploiement Amazon ECS lorsque vous souhaitez apporter et tester des modifications de service avant de les implémenter dans un environnement de production.

Prérequis

Avant de faire passer votre service du blue/green déploiement aux déploiements, assurez-vous de disposer des éléments suivants :

- Attendez que tous les déploiements en cours soient terminés.
- Un service Amazon ECS existant utilisant la stratégie de déploiement continu.
- Si plusieurs révisions de service desservent le trafic, Amazon ECS tente de consolider le trafic en une seule révision lors de la migration. En cas d'échec, vous devrez peut-être mettre à jour manuellement votre service pour utiliser une seule révision avant de procéder à la migration.
- Configurez les autorisations appropriées.
  - Pour plus d'informations sur les autorisations d'Elastic Load Balancing, consultezRôle IAM de l'infrastructure Amazon ECS pour les équilibreurs de charge.
  - Pour plus d'informations sur les autorisations Lambda, consultez. Autorisations requises pour les fonctions Lambda dans les déploiements Amazon ECS blue/green
- En fonction de la configuration, vous devez effectuer l'une des opérations suivantes :
  - Si votre service utilise Elastic Load Balancing, mettez-le à jour avec la nouvelle `AdvancedConfiguration` et lancez un déploiement progressif.
  - Si votre service utilise Service Connect, mettez-le à jour et lancez un déploiement continu.
  - Si votre service utilise à la fois Elastic Load Balancing et Service Connect, effectuez les deux étapes ci-dessus (vous pouvez utiliser une seule UpdateService requête).
  - Si votre service n'utilise aucune des solutions ci-dessus, aucune opération supplémentaire n'est nécessaire.
- Les blue/green déploiements Amazon ECS nécessitent que votre service utilise l'une des fonctionnalités suivantes. Configurez les ressources appropriées.
  - Application Load Balancer - Pour plus d'informations, consultez. Ressources d'Application Load Balancer pour les déploiements blue/green
  - Network Load Balancer - Pour plus d'informations, consultez. Ressources Network Load Balancer pour les déploiements bleu/vert d'Amazon ECS

- Service Connect - Pour plus d'informations, consultezRessources Service Connect pour les déploiements bleu/vert d'Amazon ECS.

Procédure

1. Ouvrez la console Amazon ECS à l'adressehttps://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster qui contient le service que vous souhaitez migrer.

   La page des détails du cluster s'affiche.
4. Sur la page des détails du cluster, choisissez l'onglet Services.
5. Choisissez le service, puis choisissez Mettre à jour.

   La page du service de mise à jour s'affiche
6. Développez les options de déploiement, puis effectuez les opérations suivantes :
7. Pour la stratégie de déploiement, choisissez Bleu/vert.
8. Configurez les paramètres de blue/green déploiement :

   a. Pour le temps de cuisson, entrez le nombre de minutes pendant lesquelles les révisions du service bleu et vert seront exécutées simultanément avant que la révision bleue ne soit terminée.

      Cela laisse le temps de procéder à des vérifications et à des tests.

   b. (Facultatif) Configurez les fonctions Lambda pour qu'elles s'exécutent à des étapes spécifiques du déploiement. Dans Deployment Lifecycle hooks, configurez les fonctions Lambda pour les étapes suivantes :

      - Pré-mise à l'échelle : s'exécute avant d'étendre la révision du service écologique
      - Après la mise à l'échelle : s'exécute après l'extension de la révision du service écologique
      - Tester le transfert du trafic : s'exécute pendant le routage du trafic de test vers la révision du service vert
      - Transfert de trafic après le test : s'exécute une fois que le trafic de test est acheminé vers la version verte du service
      - Transfert du trafic de production : s'exécute pendant le routage du trafic de production vers la révision du service écologique

- Réorientation du trafic après la production : s'exécute une fois que le trafic de production est acheminé vers la révision du service écologique

Pour ajouter un crochet de cycle de vie :

a. Choisissez Ajouter.

b. Pour la fonction Lambda, entrez le nom ou l'ARN de la fonction.

c. Pour Rôle, choisissez le rôle IAM autorisé à appeler la fonction Lambda.

d. Pour les étapes du cycle de vie, sélectionnez les étapes pendant lesquelles la fonction Lambda doit être exécutée.

e. Facultatif : pour les détails du crochet, entrez des paires clé-valeur pour fournir des informations supplémentaires au crochet.

9. Configurez les paramètres de l'équilibreur de charge :

   a. Sous Équilibrage de charge, vérifiez que votre service est configuré pour utiliser un équilibreur de charge.

   b. Pour Groupe cible, choisissez le groupe cible principal pour votre environnement de production (bleu).

   c. Pour Groupe cible alternatif, choisissez le groupe cible pour votre environnement de test (vert).

   d. Pour Règle d'écouteur de production, choisissez la règle d'écouteur pour acheminer le trafic de production.

   e. Facultatif : pour la règle d'écoute de test, choisissez une règle d'écoute pour acheminer le trafic de test vers votre environnement vert.

   f. Pour Rôle, choisissez le rôle IAM qui permet à Amazon ECS de gérer votre équilibreur de charge.

10. Passez en revue les modifications apportées à la configuration, puis choisissez Mettre à jour.

Étapes suivantes

- Mettez à jour le service pour démarrer le déploiement. Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.
- Surveillez le processus de déploiement pour vous assurer qu'il suit le blue/green modèle :
  - La révision du service écologique est créée et étendue

- Le trafic de test est acheminé vers la version verte (si elle est configurée)
- Le trafic de production passe à la version verte
- Après le temps de cuisson, la révision bleue est terminée

Mise à jour de la stratégie de déploiement depuis Amazon ECS blue/green jusqu'à la mise à jour continue

Vous pouvez migrer un blue/green déploiement vers un déploiement de mise à jour continue.

Tenez compte des considérations suivantes lors de la migration vers des déploiements progressifs :

- Gestion du trafic : avec les déploiements progressifs, les nouvelles tâches commencent à recevoir du trafic dès qu'elles passent les tests de santé. Il n'y a pas de phase de test distincte comme pour les blue/green déploiements.
- Efficacité des ressources : les déploiements progressifs utilisent généralement moins de ressources que blue/green les déploiements car ils remplacent les tâches progressivement au lieu de créer un environnement totalement dupliqué.
- Complexité du rollback : les déploiements progressifs compliquent les rollbacks par rapport aux déploiements. blue/green Si vous devez revenir en arrière, vous devez lancer un nouveau déploiement avec la définition de tâche précédente.
- Vitesse de déploiement : les déploiements progressifs peuvent prendre plus de temps que blue/green les déploiements, en particulier pour les services comportant de nombreuses tâches.
- Configuration de l'équilibreur de charge : votre configuration d'équilibreur de charge existante continuera de fonctionner avec des déploiements progressifs, mais le comportement de transfert du trafic sera différent.

Prérequis

Avant de migrer votre service blue/green vers des déploiements progressifs, assurez-vous de disposer des éléments suivants :

- Un service Amazon ECS existant utilisant la stratégie blue/green de déploiement
- Aucun déploiement en cours pour le service (attendez que les déploiements en cours soient terminés)
- Une compréhension claire de la façon dont votre service se comportera lors de déploiements progressifs

> **Note**
>
> Vous ne pouvez pas migrer un service vers un déploiement continu s'il est en cours de déploiement. Attendez que tous les déploiements en cours soient terminés avant de continuer.

Procédure de migration

Suivez ces étapes pour migrer votre service Amazon ECS blue/green vers des déploiements progressifs :

1. Ouvrez la console Amazon ECS à l'adressehttps://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster qui contient le service que vous souhaitez migrer.
4. Sur la page des détails du cluster, choisissez l'onglet Services.
5. Sélectionnez le service que vous souhaitez migrer, puis choisissez Mettre à jour.
6. Sur la page du service de mise à jour, accédez à la section Options de déploiement et développez-la si nécessaire.
7. Pour Stratégie de déploiement, choisissez Rolling update.
8. Configurez les paramètres de déploiement progressif :
   a. Pour Pourcentage sain minimum, entrez le pourcentage minimum de tâches qui doivent rester dans `RUNNING` cet état pendant un déploiement. Cette valeur est spécifiée sous forme de pourcentage du nombre de tâches souhaité pour le service.
   b. Dans Pourcentage maximal, entrez le pourcentage maximal de tâches autorisées à l'`PENDING`état `RUNNING` ou lors d'un déploiement. Cette valeur est spécifiée sous forme de pourcentage du nombre de tâches souhaité pour le service.
9. Facultatif : sous Détection des échecs de déploiement, configurez la manière dont Amazon ECS détecte et gère les échecs de déploiement :
   a. Pour activer le disjoncteur de déploiement, choisissez Utiliser le disjoncteur de déploiement.
   b. Pour annuler automatiquement les déploiements ayant échoué, choisissez Rollback en cas d'échec.
10. Passez en revue vos modifications de configuration, puis choisissez Mettre à jour pour enregistrer vos modifications et migrer le service vers un déploiement continu.

Amazon ECS mettra à jour la configuration de votre service afin d'utiliser la stratégie de déploiement progressif. La prochaine fois que vous mettrez à jour votre service, celui-ci utilisera le processus de déploiement progressif.

> **Note**
>
> Lorsque vous passez d'un déploiement continu blue/green à un déploiement progressif, Amazon ECS gère la transition en :
>
> 1. Identification de la révision de service active actuelle qui dessert le trafic.
> 2. Maintien de la configuration existante de l'équilibreur de charge mais modification de la façon dont les nouveaux déploiements sont gérés.
> 3. Préparation du service pour les futurs déploiements continus.

Étapes suivantes

- Mettez à jour le service pour démarrer le déploiement. Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.

Résolution des problèmes liés aux mises à jour de la stratégie de déploiement Amazon ECS

Cette section fournit des solutions aux problèmes courants que vous pouvez rencontrer lors de la migration de stratégies de déploiement.

Révisions de services ou ensembles de tâches multiples

Les problèmes suivants concernent le fait d'avoir plusieurs révisions de service pour un déploiement.

Ensembles de tâches multiples lors de la mise à jour du contrôleur de déploiement ECS

Message d'erreur : `Updating the deployment controller is not supported when there are multiple tasksets in the service. Please ensure your service has only one taskset and try again.`

Solution : Cette erreur se produit lorsque vous tentez de modifier le type de contrôleur de déploiement d'un service comportant plusieurs ensembles de tâches actifs. Pour résoudre ce problème pour le contrôleur de `EXTERNAL` déploiement `CODE_DEPLOY` ou, procédez comme suit :

1. Vérifiez les ensembles de tâches actuels :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].taskSets"
```

2. Attendez que tous les déploiements en cours soient terminés.
3. Forcer un nouveau déploiement pour nettoyer les ensembles de tâches :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --force-new-deployment
```

4. Si nécessaire, supprimez manuellement des ensembles de tâches supplémentaires :

```
aws ecs delete-task-set --cluster your-cluster-name --service your-service-name
 --task-set task-set-id
```

5. Lorsqu'il ne reste qu'un seul ensemble de tâches, réessayez de mettre à jour le contrôleur de déploiement.

Pour de plus amples informations, veuillez consulter the section called “Contrôleurs et stratégies de déploiement de services”.

Ensemble de tâches principal manquant lors de la mise à jour `ECS` du contrôleur de déploiement

Message d'erreur : `Updating the deployment controller requires a primary taskset in the service. Please ensure your service has a primary taskset and try again.`

Solution : Cette erreur se produit lorsque vous tentez de modifier le type de contrôleur de déploiement d'un service qui ne dispose pas d'un ensemble de tâches principal. Pour résoudre ce problème :

1. Vérifiez l'état du service et les ensembles de tâches.). Si un ensemble de tâches existe dans le service, il doit être marqué comme`ACTIVE`.

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].taskSets[*].[status,id]
```

S'il n'y a aucun ensemble de tâches dans `ACTIVE` cet état, migrez le déploiement. Pour plus d'informations, consultez la section Approches de migration.

2. Si aucune tâche n'est en cours d'exécution, déployez au moins une tâche en mettant à jour le service :

```
aws ecs update-service-primary-task-set --cluster your-cluster-name --
service your-service-name --primary-task-set your-taskset-id
```

   Cela marquera la tâche (précédemment`ACTIVE`) définie dans le service comme `PRIMARY` statut.

3. Attendez que la tâche atteigne un état d'exécution stable. Vous pouvez vérifier le statut avec :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].deployments"
```

4. Une fois que le service dispose d'un ensemble de tâches principal avec des tâches en cours d'exécution, réessayez de mettre à jour le contrôleur de déploiement.

Pour de plus amples informations, veuillez consulter the section called “Contrôleurs et stratégies de déploiement de services”.

Incompatibilité entre le type de détection d'échec du déploiement et le contrôleur de déploiement

Les problèmes suivants sont liés à une incompatibilité entre le type de détection d'échec du déploiement et le contrôleur de déploiement.

Disjoncteur de déploiement avec contrôleur non ECS

Message d'erreur : `Deployment circuit breaker feature is only supported with ECS deployment controller. Update to ECS deployment controller and try again.`

Solution : Cette erreur se produit lorsque vous tentez d'activer la fonction de disjoncteur de déploiement sur un service qui n'utilise pas le contrôleur de `ECS` déploiement. Le disjoncteur de déploiement est uniquement compatible avec le contrôleur `ECS` de déploiement.

1. Vérifiez le contrôleur de déploiement actuel de votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].deploymentController"
```

2. Mettez à jour votre service pour utiliser le contrôleur ECS de déploiement :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

3. Une fois que le service utilise le contrôleur ECS de déploiement, activez le disjoncteur de déploiement :

```
aws ecs update-service --cluster your-cluster-name --
service your-service-name --deployment-configuration
 "deploymentCircuitBreaker={enable=true,rollback=true}"
```

Pour de plus amples informations, veuillez consulter the section called “ Comment le disjoncteur de déploiement détecte les défaillances”.

Annulation basée sur une alarme avec un contrôleur non ECS

Message d'erreur : `Alarm based rollback feature is only supported with ECS deployment controller. Update to ECS deployment controller and try again.`

Solution : Cette erreur se produit lors de la tentative de configuration de la restauration basée sur les alarmes sur un service qui n'utilise pas le ECS contrôleur de déploiement. La fonction de restauration basée sur les alarmes est uniquement compatible avec le ECS contrôleur de déploiement.

1. Vérifiez le contrôleur de déploiement actuel de votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].deploymentController"
```

2. Mettez à jour votre service pour utiliser le contrôleur ECS de déploiement :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

3. Une fois que le service utilise le contrôleur de ECS déploiement, configurez la restauration basée sur les alarmes :

```
aws ecs update-service --cluster your-cluster-name --services your-
service-name --deployment-configuration "alarms={alarmNames=[your-alarm-
name],enable=true,rollback=true}"
```

Pour de plus amples informations, veuillez consulter the section called “Comment les CloudWatch alarmes détectent les échecs de déploiement”.

Discordance entre Service Connect et le contrôleur de déploiement

Les problèmes suivants sont liés à une incompatibilité entre Service Connect et le contrôleur de déploiement.

`EXTERNAL`contrôleur avec Service Connect

Message d'erreur : `The EXTERNAL deployment controller type is not supported for services using Service Connect.`

Solution : Cette erreur se produit lorsque vous tentez d'utiliser le contrôleur de `EXTERNAL` déploiement avec un service sur lequel Service Connect est activé. Le `EXTERNAL` contrôleur n'est pas compatible avec Service Connect.

1. Vérifiez si Service Connect est activé sur votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].serviceConnectConfiguration"
```

2. Si vous devez utiliser le contrôleur de `EXTERNAL` déploiement, désactivez Service Connect en mettant à jour votre service :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --service-connect-configuration "{}"
```

3. Sinon, si vous devez utiliser Service Connect, utilisez plutôt le contrôleur de `ECS` déploiement :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

Pour de plus amples informations, veuillez consulter the section called “Contrôleurs et stratégies de déploiement de services”.

Service Connect avec un contrôleur non ECS

Message d'erreur : `Service Connect feature is only supported with ECS (rolling update) deployment controller. Update to ECS deployment controller and try again.`

Solution : Cette erreur se produit lorsque vous tentez d'activer Service Connect sur un ordinateur qui n'utilise pas le contrôleur de `ECS` déploiement. La fonctionnalité Service Connect est uniquement compatible avec le contrôleur `ECS` de déploiement.

1. Vérifiez le contrôleur de déploiement actuel de votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].deploymentController"
```

2. Mettez à jour votre service pour utiliser le contrôleur de déploiement ECS :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

3. Une fois que le service utilise le contrôleur de déploiement ECS, activez Service Connect :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --service-connect-configuration "enabled=true,namespace=your-namespace"
```

Pour de plus amples informations, veuillez consulter the section called “Contrôleurs et stratégies de déploiement de services”.

Incompatibilité entre le type de contrôleur et la stratégie de planification

Les problèmes suivants concernent une inadéquation entre le type de contrôleur et la stratégie de planification.

`CODE_DEPLOY`contrôleur avec stratégie `DAEMON` de planification

Message d'erreur : `The CODE_DEPLOY deployment controller type is not supported for services using the DAEMON scheduling strategy.`

Solution : Cette erreur se produit lorsque vous tentez d'utiliser le contrôleur de déploiement CODE_DEPLOY avec un service utilisant la stratégie de DAEMON planification. Le CODE_DEPLOY contrôleur est uniquement compatible avec la stratégie REPLICA de planification.

1. Vérifiez la stratégie de planification actuelle de votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].schedulingStrategy"
```

2. Si vous avez besoin de blue/green déploiements, modifiez votre service pour utiliser la stratégie de REPLICA planification :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --scheduling-strategy REPLICA
```

3. Sinon, si vous devez utiliser la stratégie DAEMON de planification, utilisez plutôt le contrôleur de ECS déploiement :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

Pour de plus amples informations, veuillez consulter the section called “Contrôleurs et stratégies de déploiement de services”.

Contrôleur EXTERNE avec stratégie de planification DAEMON

Message d'erreur : `The EXTERNAL deployment controller type is not supported for services using the DAEMON scheduling strategy.`

Solution : Cette erreur se produit lorsque vous tentez d'utiliser le contrôleur de déploiement EXTERNE avec un service ECS qui utilise la stratégie de planification DAEMON. Le contrôleur EXTERNE est uniquement compatible avec la stratégie de planification REPLICA.

1. Vérifiez la stratégie de planification actuelle de votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].schedulingStrategy"
```

2. Si vous devez utiliser le contrôleur de EXTERNAL déploiement, modifiez votre service pour utiliser la stratégie REPLICA de planification :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --scheduling-strategy REPLICA
```

3. Sinon, si vous devez utiliser la stratégie DAEMON de planification, utilisez plutôt le contrôleur de ECS déploiement :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

Pour de plus amples informations, veuillez consulter the section called “Contrôleurs et stratégies de déploiement de services”.

Registres de services avec type de lancement externe

Message d'erreur : `Service registries are not supported for external launch type.`

Solution : Cette erreur se produit lors de la tentative de configuration de la découverte de services (registres de services) pour un service utilisant le type de EXTERNAL lancement. La découverte de services n'est pas compatible avec le type de EXTERNAL lancement.

1. Vérifiez le type de lancement actuel de votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].launchType"
```

2. Si vous avez besoin de découvrir un service, modifiez votre service pour utiliser le type de FARGATE lancement EC2 ou le type de lancement :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --launch-type FARGATE
```

3. Sinon, si vous devez utiliser le type de EXTERNAL lancement, supprimez la configuration du registre du service :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --service-registries "[]"
```

Pour de plus amples informations, veuillez consulter the section called “Contrôleurs et stratégies de déploiement de services”.

Annuler une mise à jour du contrôleur de déploiement

Si vous décidez de revenir au contrôleur de déploiement précédent, vous pouvez effectuer l'une des opérations suivantes :

- Si vous l'avez utilisé AWS CloudFormation, vous pouvez utiliser le modèle précédent pour créer une nouvelle pile. Pour plus d'informations, consultez la section Créer une pile à partir du Guide de AWS CloudFormation l'utilisateur.
- Si vous avez utilisé la console Amazon ECS, ou la AWS CLI, vous pouvez mettre à jour le service. Pour de plus amples informations, veuillez consulter the section called “Mise à jour d'un service”.

  Si vous utilisez la commande update-service, utilisez l'`--deployment-controller`option et définissez-la sur le contrôleur de déploiement précédent.

Afficher l'historique des services à l'aide des déploiements de services Amazon ECS

Les déploiements de services fournissent une vue complète de vos déploiements. Les déploiements de services fournissent les informations suivantes sur le service :

- La configuration de la charge de travail actuellement déployée (la révision du service source)
- Configuration de la charge de travail déployée (révision du service cible)
- État du déploiement
- Le nombre de tâches échouées détectées par le disjoncteur
- Les CloudWatch alarmes qui sont en alarme
- Quand le déploiement du service a commencé et s'est terminé
- Les détails d'une annulation, le cas échéant

Pour plus d'informations sur les propriétés de déploiement du service, consultezPropriétés incluses dans le déploiement d'un service Amazon ECS.

Les déploiements de services sont en lecture seule et chacun possède un identifiant unique.

Il existe trois étapes de déploiement du service :

Étape	Définition	États associés
En attente	Un déploiement de service a été créé, mais n'a pas démarré	PENDING
En continu	Un déploiement de service est en cours	• EN_COURS • STOP_DEMANDÉ • ROLLBACK_REQUESTED • ROLLBACK_IN_PROGRESS
Terminé	Le déploiement d'un service est terminé (avec ou sans succès)	• SUCCESSFUL • STOPPED • ROLLBACK_SUCCESSFUL • ROLLBACK_FAILED

Vous utilisez les déploiements de services pour comprendre le cycle de vie de votre service et pour déterminer si des mesures doivent être prises. Par exemple, en cas de rétrogradation, vous devrez peut-être étudier le déploiement du service et les événements du service.

Vous pouvez consulter l'historique le plus récent des 90 jours des déploiements créés le 25 octobre 2024 ou après cette date à l'aide de la console, de l'API et du. AWS CLI

Vous pouvez arrêter un déploiement qui n'est pas terminé. Pour de plus amples informations, veuillez consulter Arrêt des déploiements de services Amazon ECS.

Cycle de vie du déploiement des services

Amazon ECS crée automatiquement un nouveau déploiement de service lorsque l'une des actions suivantes se produit :

- Un utilisateur crée un service.
- Un utilisateur met à jour le service et utilise l'option Forcer un nouveau déploiement.
- Un utilisateur met à jour une ou plusieurs propriétés de service qui nécessitent un déploiement.

Pendant qu'un déploiement est en cours, Amazon ECS met à jour les propriétés de déploiement de service suivantes afin de refléter la progression du déploiement du service :

- L'état
- Le nombre de tâches en cours

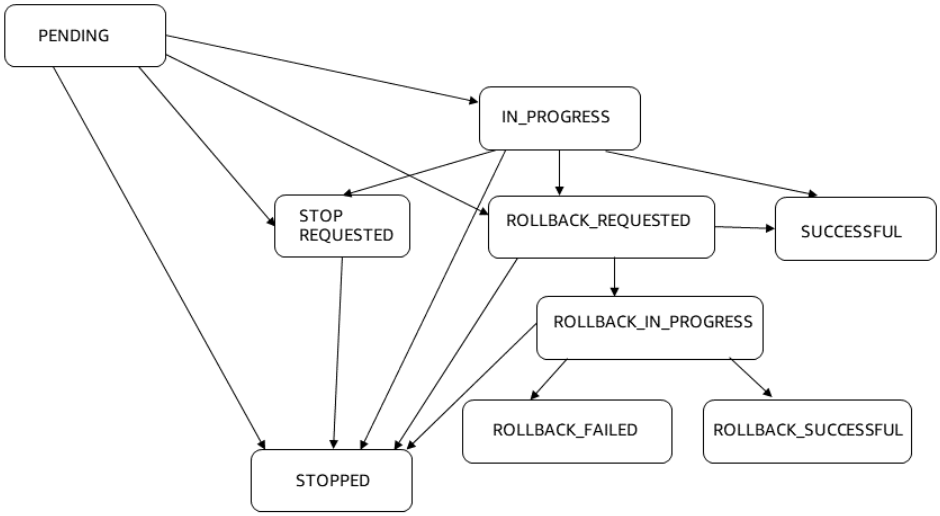
  Le nombre de tâches en cours indiqué dans la révision du service peut ne pas être égal au nombre réel de tâches en cours d'exécution. Ce nombre représente le nombre de tâches en cours d'exécution une fois le déploiement terminé. Par exemple, si vous avez lancé des tâches indépendamment du déploiement du service, ces tâches ne sont pas incluses dans le nombre de tâches en cours pour la révision du service.
- Détection de panne du disjoncteur :
  - Le nombre de tâches qui n'ont pas pu démarrer
- CloudWatch détection de panne d'alarme
  - Les alarmes actives
- Informations relatives à l'annulation :
  - L'heure de début
  - La raison du rollback
  - L'ARN de la révision du service utilisée pour le rollback
- La raison du statut

Amazon ECS supprime le déploiement du service lorsque vous supprimez un service.

États de déploiement des services

Le déploiement d'un service démarre en `PENDING` état.

L'illustration suivante montre les états de déploiement du service qui peuvent se produire après l'`PENDING`état : `IN_PROGRESS` `ROLLBACK_REQUESTED``SUCCESSFUL`,`STOP_REQUESTED`,`ROLLBACK_IN_PROGRESS`,`ROLLBACK_FAILED`, et`STOPPED`.



Les informations suivantes fournissent des informations détaillées sur les états de déploiement des services :

- `PENDING`- Le déploiement du service a été créé, mais n'a pas démarré.

  L'État peut se déplacer vers `IN_PROGRESS``ROLLBACK_REQUESTED`,`STOP_REQUESTED`, ou`STOPPED`.

- `IN_PROGRESS`- Le déploiement du service est en cours.

  L'État peut se déplacer vers `SUCCESSFUL``STOP_REQUESTED`,`ROLLBACK_REQUESTED`,`ROLLBACK_IN_PROGRESS`, et`STOPPED`.

- `STOP_REQUESTED`- L'état de déploiement du service passe au `STOP_REQUESTED` moment où l'une des situations suivantes se produit :
  - Un utilisateur lance le déploiement d'un nouveau service.
  - L'option de restauration n'est pas utilisée pour le mécanisme de détection des défaillances (disjoncteur ou alarme) et le service n'atteint pas son état normal. `SUCCESSFUL`

  L'État passe à`STOPPED`.

- `ROLLBACK_REQUESTED`- L'état de déploiement du service passe au `ROLLBACK_REQUESTED` moment où un utilisateur demande une annulation par le biais de la console, de l'API ou de la CLI.

  L'État peut se déplacer vers `SUCCESSFUL``ROLLBACK_IN_PROGRESS`, et`STOPPED`.
- `SUCCESSFUL`- L'état de déploiement du service passe au `SUCCESSFUL` moment où le déploiement du service est terminé avec succès.
- `ROLLBACK_IN_PROGRESS`- L'état de déploiement du service passe au `ROLLBACK_IN_PROGRESS` moment où l'option de restauration est utilisée pour le mécanisme de détection des défaillances (disjoncteur ou alarme) et que le service échoue.

  L'État se déplace vers`ROLLBACK_SUCCESSFUL`, ou`ROLLBACK_FAILED`.

Propriétés incluses dans le déploiement d'un service Amazon ECS

Les propriétés suivantes sont incluses dans le déploiement d'un service.

Propriété	Description
ARN de déploiement des services	L'ARN du déploiement du service.
ARN du service	L'ARN du service pour ce déploiement de service.
ARN de cluster	L'ARN du cluster qui héberge le service.
Heure de création du déploiement du service	Heure à laquelle le déploiement du service a été créé.
Heure de début du déploiement du service	Heure à laquelle le déploiement du service a commencé.
Heure de fin du déploiement du service	Heure à laquelle le déploiement du service s'est terminé.
Heure d'arrêt du déploiement du service	Heure à laquelle le déploiement du service s'est arrêté.

Propriété	Description	
Heure de mise à jour du déploiement du service	Heure à laquelle le déploiement du service a été mis à jour pour la dernière fois.	
Révisions du service source	Les révisions du service en cours. Pour plus d'informations sur les propriétés incluses, consultez Propriétés incluses dans une révision du service Amazon ECS.	

Propriété	Description	
Configuration de déploiement	Les paramètres de déploiement, y compris la configuration du disjoncteur, les alarmes qui le déterminent. • Limite inférieure du nombre de tâches qui doivent être exécutées pour un service lors d'un déploiement ou lorsqu'une instance de conteneur est épuisée, en pourcentage du nombre de tâches souhaité pour le service. • Limite supérieure du nombre de tâches qui doivent être exécutées pour un service lors d'un déploiement ou lorsqu'une instance de conteneur est épuisée, en pourcentage du nombre de tâches souhaité pour le service. • La configuration du disjoncteur. • Les alarmes utilisées pour détecter les défaillances.	

Propriété	Description
Révision du service cible	Révision du service à déployer. Une fois le déploiement terminé avec succès, la révision du service cible est la révision du service en cours d'exécution.
État du déploiement du service	État de déploiement du service. Les valeurs valides sont PENDING, SUCCESSFUL, STOPPED, STOP_REQU ESTED, STOP_IN_P ROGRESS, IN_PROGRESS, ROLLBACK_IN_PROGRESS, ROLLBACK_SUCCESSFUL et ROLLBACK_FAILED.
Informations sur l'état du déploiement du service	Informations expliquant pourquoi le déploiement du service est dans l'état actuel. Par exemple, le disjoncteur a détecté une panne.
Informations de rétrogradation	Les options de restauration utilisées par le déploiement du service en cas d'échec du déploiement.
Options de disjoncteur pour le déploiement du service	Le disjoncteur qui détermine le déploiement d'un service a échoué.

Propriété	Description	
CloudWatch alarmes pour le déploiement du service	Les CloudWatch alarmes qui déterminent l'échec du déploiement d'un service.	

Autorisations requises pour consulter les déploiements de services Amazon ECS

Lorsque vous suivez la meilleure pratique consistant à accorder le moindre privilège, vous devez ajouter des autorisations supplémentaires afin de visualiser les déploiements de services dans la console.

Vous devez avoir accès aux actions suivantes :

- ListServiceDeployments
- DescribeServiceDeployments
- DescribeServiceRevisions

Vous devez avoir accès aux ressources suivantes :

- Service
- Déploiement du service
- Révision du service

L'exemple de politique suivant contient les autorisations requises et limite les actions à un service spécifique.

Remplacez le `accountcluster-name`, et `service-name` par vos valeurs.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "ecs:ListServiceDeployments",
              "ecs:DescribeServiceDeployments",
              "ecs:DescribeServiceRevisions"
```

```
            ],
            "Resource": [
             "arn:aws:ecs:us-east-1:123456789012:service/cluster-name/service-name",
             "arn:aws:ecs:us-east-1:123456789012:service-deployment/cluster-name/
service-name/*",
             "arn:aws:ecs:us-east-1:123456789012:service-revision/cluster-name/service-
name/*"
            ]
        }
    ]
}
```

Affichage des déploiements de services Amazon ECS

Vous pouvez consulter l'historique des 90 jours le plus récent pour les déploiements créés le 25 octobre 2024 ou après cette date. Les déploiements de services peuvent se faire dans l'un des états suivants :

- En cours
- En attente
- Terminé

Vous pouvez utiliser ces informations pour déterminer si vous devez mettre à jour la manière dont le service est déployé ou la révision du service. Pour plus d'informations sur les propriétés incluses, consultez Propriétés incluses dans le déploiement d'un service Amazon ECS.

Avant de commencer, configurez les autorisations requises pour visualiser les déploiements de services. Pour de plus amples informations, veuillez consulter Autorisations requises pour consulter les déploiements de services Amazon ECS.

Amazon ECS Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page des détails du service s'affiche.
4. Sur la page des détails du service, sélectionnez Déploiements.
5. Choisissez le déploiement du service à afficher.

Pour afficher les déploiements de services pour ce type de déploiement	Faites ceci
Déploiement en cours	Sous Déploiements en cours, choisissez l'ID de déploiement.
Dernier déploiement	Sous Dernier déploiement, choisissez l'ID de déploiement.
Déploiements terminés	Sous Déploiements de services, choisissez l'ID de déploiement.

La page des détails du déploiement du service apparaît.

6. (Facultatif) Comparez les révisions des services pour voir les différences.

   Sous Révisions du service, choisissez Comparer les révisions, puis sélectionnez 2 révisions à comparer.

   Les révisions du service sont affichées side-by-side avec les différences mises en évidence.

AWS CLI

1. Exécutez `list-service-deployments` pour récupérer l'ARN de déploiement du service.

   Remplacez les variables par vos valeurs.

```
aws ecs list-service-deployments --cluster cluster-name --service service-name
```

   Notez serviceDeploymentArn le déploiement que vous souhaitez consulter.

```
{
    "serviceDeployments": [
        {
```

```
            "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-
deployment/example/sd-example/NCWGC2ZR-taawPAYrIaU5",
            "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/example/
sd-example",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/example",
            "targetServiceRevisionArn": "arn:aws:ecs:us-
west-2:123456789012:service-revision/example/sd-example/4980306466373577095",
            "status": "SUCCESSFUL"
        }
    ]
}
```

2. Exécutez `describe-service-deployments`. Utilisez celui `serviceDeploymentArn` qui a été renvoyé`list-service-deployments`.

   Remplacez les variables par vos valeurs.

```
aws ecs describe-service-deployments --service-deployment-arns
 arn:aws:ecs:region:123456789012:service-deployment/cluster-name/service-
name/NCWGC2ZR-taawPAYrIaU5
```

Étapes suivantes

Vous pouvez consulter les détails des révisions de service lors du déploiement. Pour de plus amples informations, consultez Afficher les détails de révision du service Amazon ECS.

Révisions du service Amazon ECS

Une révision de service contient un enregistrement de la configuration de charge de travail qu'Amazon ECS tente de déployer. Chaque fois que vous créez ou déployez un service, Amazon ECS crée et capture automatiquement la configuration que vous essayez de déployer dans le cadre de la révision du service.

Les révisions de service sont en lecture seule et comportent des identifiants uniques. Pour plus d'informations sur les propriétés incluses, consultezPropriétés incluses dans une révision du service Amazon ECS.

Les révisions de service offrent les avantages suivants :

- Lors d'un déploiement de service, vous pouvez comparer la révision du service actuellement déployée (source) avec celle en cours de déploiement (cible).

- Lorsque vous utilisez l'option d'annulation pour un déploiement de service, Amazon ECS annule automatiquement le déploiement du service jusqu'à la dernière révision de service déployée avec succès.
- Les révisions de service contiennent l'enregistrement de la configuration de la charge de travail dans une ressource.

Cycle de vie de révision des services

Amazon ECS crée automatiquement une nouvelle révision de service lorsque vous créez un service ou que vous mettez à jour une propriété de service qui lance un déploiement.

Amazon ECS ne crée pas de nouvelle révision de service pour une opération de restauration. Amazon ECS utilise la dernière révision de service réussie pour l'annulation.

La révision d'un service est immuable.

Amazon ECS supprime la révision du service lorsque vous supprimez un service.

Vous pouvez consulter les révisions de service créées le 25 octobre 2024 ou après cette date à l'aide de la console, de l'API et de la CLI.

Propriétés incluses dans une révision du service Amazon ECS

Les propriétés suivantes sont incluses dans une révision de service.

Ressource	Description
ARN du service	ARN qui identifie le service.
ARN de cluster	L'ARN du cluster qui héberge le service.
ARN de définition de tâche	L'ARN de la définition de tâche utilisée pour les tâches de service.
Registres de services	Les détails des registres de services utilisés pour la découverte des services.

Ressource	Description	
	• Le nom du conteneur La valeur du nom du conteneur à utiliser pour votre service de découverte de service. • Le port à conteneurs La valeur du port à utiliser pour votre service de découverte de service. Cette valeur est également spécifiée dans la définition de la tâche. • Le port Valeur de port utilisée si votre service de découvert e de service a spécifié un registre SRV. • L'ARN du registre L'Amazon Resource Name (ARN) du registre de service.	

Ressource	Description	
Fournisseurs de capacité	Détails de la stratégie du fournisseur de capacité. • Le nom du fournisseur de capacité Le nom du fournisseur de capacité utilisé pour le service. • Le poids du fournisseur de capacité Pourcentage relatif du nombre total de tâches lancées qui devraient utiliser le fournisseur de capacité spécifié. • La base de fournisseurs de capacité Le nombre minimum de tâches à exécuter sur le fournisseur de capacité spécifié.	
Images de conteneur	Les détails concernant les images du conteneur. • Le nom du conteneur • Le condensé d'images • L'image du conteneur	

Ressource	Description
Réseaux	Configuration réseau du service. • S'il existe une adresse IP publique • Les sous-réseaux dans lesquels les tâches s'exécutent • Les groupes de sécurité qui contrôlent le trafic des services
Type de lancement	Type de lancement utilisé pour le service.

Propriétés spécifiques à Fargate	Lorsque le type de lancement est Fargate, il s'agit d'informa tions sur la version de Fargate.
	• La version de plateforme et la famille de plateformes
	• Stockage éphémère
	  • La quantité de stockage éphémère à allouer pour le déploiement.
	  • La clé KMS à utiliser pour le chiffrement du stockage.

Ressource	Description	
Volumes Amazon EBS configurés lors du déploiement	Configuration d'un volume spécifié dans la définition de tâche en tant que volume configuré au moment du lancement. • Chiffré Indique si le volume est chiffré. • Type de système de fichiers Type de système de fichiers Linux pour le volume. • IOPS Nombre d' I/O opérations par seconde (IOPS). • Clé KMS • Rôle IAM L'ARN du rôle IAM à associer à ce volume. • Size La taille du volume en Gio. • ID de l'instantané ID de capture d'écran du volume. • Spécifications du tag Configuration des balises pour le volume. • Débit	

Ressource	Description
	Débit provisionné pour le volume. • Type de volume Type de volume.
Service Connect	La configuration de Service Connect. • L'indication indiquant si Service Connect est en cours d'utilisation • L'espace de noms • La liste des services interconnectés • Configuration de journalisation de Service Connect
Équilibreurs de charge de service	Les équilibreurs de charge qui acheminent le trafic de service. • Le nom • Le nom du conteneur • Le port à conteneurs • L'ARN du groupe cible
Surveillance d'exécution	Indique si la surveillance du temps d'exécution est activée.
Date de création	Date à laquelle la révision du service a été créée.
VPC Lattice	Configuration du réseau VPC pour la révision du service.

Afficher les détails de révision du service Amazon ECS

Vous pouvez consulter les informations relatives aux types de révisions de service suivants qui ont été créés le 25 octobre 2024 ou après cette date :

- Source -Configuration de la charge de travail actuellement déployée
- Cible : configuration de la charge de travail en cours de déploiement

Amazon ECS Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page des détails du service s'affiche.
4. Sur la page des détails du service, sélectionnez Déploiements.
5. Choisissez la version du service à afficher.

Pour consulter les révisions du service pour ce type de déploiement	Faites ceci	
Déploiements en cours	Sous Déploiements en cours, procédez comme suit : • Pour consulter la révision du service source, sous Révisions du service, choisissez l'ID de révision du service pour le type de révision source. • Pour afficher la révision du service cible, sous Révisions du service, choisissez l'ID de révision	

Pour consulter les révisions du service pour ce type de déploiement	Faites ceci
	du service pour le type de révision cible.
Dernier déploiement	Sous Dernier déploiement, choisissez la révision du service cible.
Déploiements terminés	Sous Déploiements de services, procédez comme suit : • Sous Révision du service Target, choisissez l'ID.

AWS CLI

1. Exécutez `describe-service-deployments` pour récupérer l'ARN de révision du service.

   Remplacez les variables par vos valeurs.

```
aws ecs describe-service-deployments --service-deployment-arns
 arn:aws:ecs:region:account-id:service/cluster-name/service-name/NCWGC2ZR-
taawPAYrIaU5
```

   Notez le `arn` pour `sourceServiceRevisions` ou le`targetServiceRevisions`.

```
{
    "serviceDeployments": [
        {
            "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-
deployment/example/sd-example/NCWGC2ZR-taawPAYrIaU5",
            "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/example/
sd-example",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/example",
            "updatedAt": "2024-09-10T16:49:35.572000+00:00",
            "sourceServiceRevision": {
```

```
                "arn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
example/sd-example/4980306466373578954",
                "requestedTaskCount": 0,
                "runningTaskCount": 0,
                "pendingTaskCount": 0
            },
            "targetServiceRevision": {
                "arn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
example/sd-example/4980306466373577095",
                "requestedTaskCount": 0,
                "runningTaskCount": 0,
                "pendingTaskCount": 0
            },
            "status": "IN_PROGRESS",
            "deploymentConfiguration": {
                "deploymentCircuitBreaker": {
                    "enable": false,
                    "rollback": false
                },
                "maximumPercent": 200,
                "minimumHealthyPercent": 100
            }
        }
    ],
    "failures": []
}
```

2. Exécutez `describe-service-revisions`. Utilisez celui `arn` qui a été renvoyé`describe-service-deployments`.

   Remplacez les variables par vos valeurs.

```
aws ecs describe-service-revisions --service-revision-arns
 arn:aws:ecs:region:123456789012:service-revision/cluster-name/service-
name/4980306466373577095
```

Équilibrer un service Amazon ECS entre les zones de disponibilité

Pour aider vos applications à atteindre une haute disponibilité, nous vous recommandons de configurer vos services multitâches pour qu'ils s'exécutent sur plusieurs zones de disponibilité. Pour les services dont la première stratégie de placement est de répartir les zones de disponibilité, AWS

faites de votre mieux pour répartir uniformément les tâches de service entre les zones de disponibilité disponibles.

Cependant, il peut arriver que le nombre de tâches exécutées dans une zone de disponibilité ne soit pas le même que dans d'autres zones de disponibilité, par exemple après une interruption de zone de disponibilité. Pour remédier à ce déséquilibre des tâches, vous pouvez activer la fonction de rééquilibrage des zones de disponibilité.

Grâce au rééquilibrage des zones de disponibilité, Amazon ECS surveille en permanence la répartition des tâches entre les zones de disponibilité pour chacun de vos services. Lorsqu'Amazon ECS détecte une répartition inégale des tâches, il prend automatiquement des mesures pour rééquilibrer la charge de travail entre les zones de disponibilité. Cela implique de lancer de nouvelles tâches dans les zones de disponibilité comportant le moins de tâches et de terminer des tâches dans les zones de disponibilité surchargées.

Cette redistribution garantit qu'aucune zone de disponibilité ne devienne un point de défaillance, ce qui contribue à maintenir la disponibilité globale de vos applications conteneurisées. Le processus de rééquilibrage automatisé élimine le besoin d'intervention manuelle, accélérant ainsi le temps de rétablissement après un événement.

Voici un aperçu du processus de rééquilibrage des zones de disponibilité :

1. Amazon ECS commence à surveiller un service une fois qu'il a atteint l'état stable et examine le nombre de tâches exécutées dans chaque zone de disponibilité.
2. Amazon ECS effectue les opérations suivantes lorsqu'il détecte un déséquilibre dans le nombre de tâches exécutées dans chaque zone de disponibilité :
   - Envoie un événement de service indiquant que le rééquilibrage de la zone de disponibilité commence.
   - Démarre les tâches dans les zones de disponibilité avec le plus petit nombre de tâches en cours
   - Arrête les tâches dans les zones de disponibilité comportant le plus grand nombre de tâches en cours d'exécution.
   - Le planificateur attend que les tâches nouvellement démarrées soient terminées `HEALTHY` et `RUNNING` avant de les arrêter dans la zone de disponibilité surdimensionnée.
   - Envoie un événement de service avec le résultat du rééquilibrage de la zone de disponibilité.

Comment Amazon ECS détecte une répartition inégale des tâches

Amazon ECS détermine un déséquilibre dans le nombre de tâches exécutées dans chaque zone de disponibilité en divisant le nombre de tâches souhaité pour le service par le nombre de zones de disponibilité configurées. Si le nombre de tâches souhaité ne se répartit pas de manière égale, Amazon ECS répartit le reste des tâches de manière égale entre les zones de disponibilité configurées. Chaque zone de disponibilité doit comporter au moins une tâche.

Prenons l'exemple d'un service Amazon ECS avec un nombre souhaité de deux tâches configurées pour deux zones de disponibilité. Dans ce scénario, le nombre de tâches souhaité est réparti de manière égale. Une distribution équilibrée serait une tâche par zone de disponibilité. S'il y a deux tâches dans la zone de disponibilité 1 et aucune tâche dans la zone de disponibilité 2, Amazon ECS initie le rééquilibrage en démarrant une tâche dans la zone de disponibilité 2 avant d'arrêter une tâche dans la zone de disponibilité 1.

À présent, imaginez un service Amazon ECS avec un nombre souhaité de trois tâches configurées pour deux zones de disponibilité. Dans ce scénario, le nombre de tâches souhaité ne se divise pas de manière égale. Une distribution équilibrée comporterait une tâche dans la zone de disponibilité 1 et deux tâches dans la zone de disponibilité 2, car chaque zone de disponibilité comporte au moins une tâche et le reste de la tâche est placée dans la zone de disponibilité 2.

Prenons l'exemple d'un service Amazon ECS dont le nombre souhaité de cinq tâches est configuré pour trois zones de disponibilité. Dans ce scénario, le nombre de tâches souhaité ne se divise pas de manière égale. Une distribution équilibrée comporterait une tâche dans la zone de disponibilité 1 et deux tâches dans chacune des zones de disponibilité 2 et 3. Après avoir pris en compte le fait que chaque zone de disponibilité comporte une tâche chacune, les deux tâches restantes sont réparties uniformément entre les zones de disponibilité.

Considérations relatives à la configuration du rééquilibrage des zones de disponibilité

Tenez compte des points suivants lorsque vous souhaitez configurer le rééquilibrage des zones de disponibilité :

- Le rééquilibrage des zones de disponibilité prend en charge les types Fargate et Launch EC2 . Pour Fargate, Amazon ECS redistribuera automatiquement les tâches entre les zones de disponibilité disponibles afin de maintenir l'équilibre. Pour le type de EC2 lancement, Amazon ECS rééquilibre les tâches entre les instances de conteneur existantes dans la mesure du possible, en respectant les stratégies et les contraintes de placement que vous avez définies. Toutefois, Amazon ECS ne peut pas lancer de nouvelles instances dans les zones de disponibilité sous-

utilisées dans le cadre du processus de rééquilibrage, ce qui limite le rééquilibrage aux instances de conteneur existantes.

- Le rééquilibrage des zones de disponibilité fonctionne dans les configurations suivantes :
  - Services qui utilisent la `Replica` stratégie
  - Services qui spécifient la répartition de la zone de disponibilité comme première stratégie de placement des tâches, ou qui ne spécifient pas de stratégie de placement.
- Vous ne pouvez pas utiliser le rééquilibrage des zones de disponibilité avec des services répondant à l'un des critères suivants :
  - Utilise la `Daemon` stratégie
  - Utilise le type de `EXTERNAL` lancement (ECS Anywhere)
  - Utilise 100 % pour la `maximumPercent` valeur
  - Utilise un Classic Load Balancer
  - Utilise le `attribute:ecs.availability-zone` comme contrainte de placement des tâches

Stratégies de placement et contraintes de placement avec rééquilibrage des zones de disponibilité

Les stratégies de placement déterminent la manière dont Amazon ECS sélectionne les instances de conteneur et les zones de disponibilité pour la fin du placement des tâches. Les contraintes de placement des tâches sont des règles qui déterminent si une tâche est autorisée à s'exécuter sur une instance de conteneur spécifique.

Pour le type de EC2 lancement, vous pouvez utiliser des stratégies de placement et des contraintes de placement en conjonction avec le rééquilibrage des zones de disponibilité. Toutefois, pour que le rééquilibrage des zones de disponibilité fonctionne, la stratégie de placement des écarts de zone de disponibilité doit être la première stratégie spécifiée.

Le rééquilibrage des zones de disponibilité est compatible avec différentes combinaisons de stratégies de placement. Par exemple, vous pouvez créer une stratégie qui répartit d'abord les tâches de manière uniforme entre les zones de disponibilité, puis qui regroupe les tâches en fonction de la mémoire au sein de chaque zone de disponibilité. Dans ce cas, le rééquilibrage des zones de disponibilité fonctionne car la stratégie de spread des zones de disponibilité est spécifiée en premier.

Il est important de noter que le rééquilibrage des zones de disponibilité ne fonctionnera pas si la première stratégie du tableau de stratégies de placement n'est pas un composant de spread des

zones de disponibilité. Cette exigence garantit que l'objectif principal de la distribution des tâches est de maintenir l'équilibre entre les zones de disponibilité, ce qui est crucial pour une haute disponibilité.

Pour plus d'informations sur les stratégies et les contraintes de placement des tâches, consultezComment Amazon ECS place les tâches sur les instances de conteneur.

Les stratégies et contraintes de placement de tâche ne sont pas prises en charge pour les tâches utilisant le type de lancement Fargate. Fargate fera de son mieux pour répartir les tâches entre les zones de disponibilité accessibles. Si le fournisseur de capacité inclut à la fois Fargate et Fargate Spot, le comportement de répartition est indépendant pour chaque fournisseur de capacité.

L'exemple de stratégie suivant répartit les tâches de manière uniforme entre les zones de disponibilité, puis regroupe les tâches en fonction de la mémoire au sein de chaque zone de disponibilité. Le rééquilibrage des zones de disponibilité est compatible avec le service car c'est la `spread` stratégie qui prime.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Activer le rééquilibrage des zones de disponibilité

Vous devez activer le rééquilibrage des zones de disponibilité pour les services nouveaux et existants.

Vous pouvez activer et désactiver le rééquilibrage des zones de disponibilité à l'aide de la console ou du AWS CLI. APIs

Type de service	« Hello, World! »	console	INTERFACE DE LIGNE DE COMMANDE (CLI)
Existant	UpdateService	Mettre à jour un service Amazon ECS	service de mise à jour
New	CreateService	Création d'un déploiement de mises à jour continues Amazon ECS	create-service

L'exemple suivant montre comment activer le rééquilibrage des services lors de la création d'un nouveau service :

```
aws ecs create-service \
    --cluster my-cluster \
    --service-name my-service \
    --task-definition my-task-definition:1 \
    --desired-count 6 \
    --availability-zone-rebalancing ENABLED
```

Suivez le rééquilibrage des zones de disponibilité d'Amazon ECS

Vous pouvez vérifier si le rééquilibrage des zones de disponibilité est activé pour un service dans la console ou en appelant`describe-services`. L'exemple suivant peut être utilisé pour voir l'état avec la CLI.

La réponse sera soit, `ENABLED` soit`DISABLED`.

```
aws ecs describe-services \
    --services service-name \
    --cluster cluster-name \
    --query services[0].availabilityZoneRebalancing
```

Événements liés au service

Amazon ECS envoie des événements d'action de service pour vous aider à comprendre le cycle de vie du rééquilibrage des zones de disponibilité.

Événement	Scénario	Type	En savoir plus
SERVICE_REBALANCING_STARTED	Amazon ECS lance une opération de rééquilibrage des zones de disponibilité	INFO	service (*service-name*) n'est pas équilibré entre les *number-tasks* tâches *number-tasks* dans *Availability Zone 1Availability Zone 2* , dans et *number-tasks* dans*Availability Zone 3* . Rééquilibrage AZ en cours.
SERVICE_REBALANCING_COMPLETED	L'opération de rééquilibrage de la zone de disponibilité est terminée	INFO	service (*service-name*) est équilibré en AZ avec *number-tasks* les tâches dans*Availability Zone 1* , *number-tasks* les tâches dans *Availability Zone 2* et *number-tasks* les tâches dans*Availability Zone 3* .
TASKS_STARTED	Amazon ECS démarre avec succès les tâches dans le cadre de l'opération de rééquilibrage de la zone de disponibilité	INFO	*service-name* a commencé *number-tasks* des tâches dans *Availability Zone* AZ

Événement	Scénario	Type	En savoir plus
			Rebalance :*task-ids*.
TASKS_STOPPED	Amazon ECS arrête avec succès les tâches dans le cadre de l'opération de rééquilibrage de la zone de disponibilité	INFO	*service-name* a cessé *number-tasks* d'exécuter des tâches en *Availability Zone* raison du rééquilibrage AZ :*task-id*.
SERVICE_TASK_PLACEMENT_FAILURE	Amazon ECS n'a pas pu démarrer une tâche dans le cadre de l'opération de rééquilibrage de la zone de disponibilité	ERROR	Pour le type de EC2 lancement, voir service (*service-name*) n'est pas en mesure de placer une tâche *Availability Zone* car aucune instance de conteneur ne répond à toutes ses exigences. Pour le type de lancement Fargate, voir service (*service-name*) ne parvient pas à placer une tâche dedans*Availability Zone* .

Événement	Scénario	Type	En savoir plus
`TASKSET_SCALE_IN_FAILURE_BY_TASK_PROTECTION`	L'opération de rééquilibrage de la zone de disponibilité est bloquée car la protection des tâches est utilisée.	INFO	service (*`service-name`*) n'a pas pu effectuer AZ Rebalance car il n'*`task-set-name`* a pas pu être intégré en raison de*`reason`*.
`SERVICE_REBALANCING_STOPPED`	L'opération de rééquilibrage de la zone de disponibilité s'est arrêtée. Amazon ECS envoie des événements supplémentaires qui fournissent des informations supplémentaires.	INFO	service (*`service-name`*) a arrêté AZ Rebalancing.

Événements de modification de l'état de la tâche

Amazon ECS envoie un événement de changement d'état de tâche (`START`) pour chaque tâche démarrée dans le cadre du processus de rééquilibrage.

Amazon ECS envoie un événement de changement d'état de tâche (`STOPPED`) pour chaque tâche arrêtée dans le cadre du processus de rééquilibrage. La raison est définie sur`Availability-zone rebalancing initiated by (deployment ecs-svc/`*`deployment-id`*`)`.

Pour plus d'informations sur les événements, consultezÉvénements de modification de l'état des tâches Amazon ECS.

Résolution des problèmes liés au rééquilibrage des services

Si vous rencontrez des problèmes lors du rééquilibrage des services, envisagez les étapes de résolution des problèmes suivantes :

Le rééquilibrage ne démarre pas

Vérifiez que :

- Le rééquilibrage des services est activé pour votre service
- Votre service utilise une configuration prise en charge (voirthe section called “Considérations relatives à la configuration du rééquilibrage des zones de disponibilité”)
- Votre service a atteint un état stable

Défaillances du placement des tâches lors du rééquilibrage

Si vous voyez `SERVICE_TASK_PLACEMENT_FAILURE` des événements :

- Pour EC2 le type de lancement : Vérifiez si des instances de conteneur sont disponibles dans la zone de disponibilité cible
- Pour le type de lancement Fargate : vérifiez s'il existe des contraintes de ressources ou des quotas de service limitant le placement des tâches
- Passez en revue vos contraintes de placement des tâches pour vous assurer qu'elles n'empêchent pas une bonne répartition des tâches

Le rééquilibrage s'arrête de façon inattendue

Si vous voyez `SERVICE_REBALANCING_STOPPED` des événements :

- Vérifiez s'il existe une protection des tâches susceptible de bloquer l'opération
- Recherchez les déploiements de services simultanés susceptibles d'interrompre le rééquilibrage
- Consultez les événements de service pour obtenir des informations supplémentaires sur les raisons pour lesquelles le rééquilibrage a été arrêté

Meilleures pratiques pour le rééquilibrage des services

Suivez ces bonnes pratiques pour tirer le meilleur parti du rééquilibrage des services :

- Surveillez les opérations de rééquilibrage - Configurez des CloudWatch alarmes pour surveiller les événements de service liés au rééquilibrage afin d'identifier rapidement tout problème.
- Tenez compte de l'impact sur les performances : sachez que les opérations de rééquilibrage peuvent augmenter temporairement l'utilisation des ressources lorsque de nouvelles tâches sont lancées avant que les anciennes ne soient arrêtées.

- Utilisez la protection des tâches de manière stratégique : si vous avez des tâches critiques qui ne doivent pas être interrompues pendant le rééquilibrage, envisagez d'utiliser la protection des tâches.
- Planification de la EC2 capacité : pour EC2 le type de lancement, assurez-vous de disposer de suffisamment d'instances de conteneurs dans toutes les zones de disponibilité pour permettre un rééquilibrage efficace.
- Testez le comportement du rééquilibrage - Avant de vous fier au rééquilibrage en production, testez le comportement de vos services lors des opérations de rééquilibrage dans un environnement hors production.

Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS

Votre service peut éventuellement être configuré pour utiliser Elastic Load Balancing afin de répartir le trafic de manière uniforme entre les tâches de votre service.

> **Note**
>
> Lorsque vous utilisez des ensembles de tâches, toutes les tâches de l'ensemble doivent être configurées pour utiliser Elastic Load Balancing ou pour ne pas utiliser Elastic Load Balancing.

Les services Amazon ECS hébergés sur ce site AWS Fargate prennent en charge les équilibreurs de charge d'application, les équilibreurs de charge réseau et les équilibreurs de charge de passerelle. Consultez le tableau suivant pour savoir quel type d'équilibreur de charge utiliser.

Type de Load Balancer	À utiliser dans ces cas	
Application Load Balancer	Trafic routier HTTP/HTTPS (ou couche 7). Les équilibreurs de charge Application Load Balancer offrent plusieurs fonctions qui les rendent intéressants	

Type de Load Balancer	À utiliser dans ces cas	
	à utiliser avec les services Amazon ECS service : • Chaque service peut desservir le trafic provenant de plusieurs équilibreurs de charge et exposer plusieurs ports à charge équilibrée en spécifiant plusieurs groupes cibles. • Ils sont pris en charge par des tâches hébergées à la fois sur EC2 Fargate et sur des instances. • Les Application Load Balancers permettent aux conteneurs d'utiliser le mappage de port hôte dynamique (de sorte que plusieurs tâches du même service soient autorisées par instance de conteneur). • Les Application Load Balancers prennent en charge le routage basé sur le chemin d'accès et les règles de priorité (pour que plusieurs services puissent utiliser le même port d'écoute sur un même Application Load Balancer).	
Network Load Balancer	Acheminez le trafic TCP ou UDP (ou couche 4).	

Type de Load Balancer	À utiliser dans ces cas	
Passerelle équilibreur de charge	Acheminez le trafic TCP ou UDP (ou couche 4). Utilisez des appareils virtuels, tels que des pare-feux, des systèmes de détection et de prévention des intrusions et des systèmes d'inspection approfondie des paquets.	

Nous vous recommandons d'utiliser des équilibreurs de charge d'application pour vos services Amazon ECS afin de tirer parti de ces dernières fonctionnalités, sauf si votre service nécessite une fonctionnalité uniquement disponible avec les équilibreurs de charge réseau ou les équilibreurs de charge de passerelle. Pour plus d'informations sur Elastic Load Balancing et les différences entre les types d'équilibreurs de charge, consultez le Guide de l'utilisateur Elastic Load Balancing.

Avec votre équilibreur de charge, vous payez uniquement en fonction de votre utilisation. Pour plus d'informations, veuillez consultez Tarification Elastic Load Balancing.

Optimisation des paramètres de vérification de l'état de l'équilibreur de charge pour Amazon ECS

Les équilibreurs de charge acheminent les demandes uniquement vers les cibles saines dans les zones de disponibilité de l'équilibreur de charge. Chaque cible est enregistrée auprès d'un groupe cible. L'équilibreur de charge vérifie l'état de santé de chaque cible à l'aide des paramètres de vérification de l'état du groupe cible. Une fois que vous avez enregistré la cible, elle doit passer un test de santé pour être considérée comme saine. Amazon ECS surveille l'équilibreur de charge. L'équilibreur de charge envoie régulièrement des bilans de santé au conteneur Amazon ECS. L'agent Amazon ECS surveille et attend que l'équilibreur de charge publie un rapport sur l'état du conteneur. Il le fait avant de considérer que le contenant est en bon état.

Deux paramètres de vérification de l'état d'Elastic Load Balancing affectent la vitesse de déploiement :

- Intervalle entre les contrôles de santé : détermine le temps approximatif, en secondes, entre les contrôles de santé d'un contenant individuel. Par défaut, l'équilibreur de charge vérifie toutes les 30 secondes.

  Ce paramètre est nommé :

  - `HealthCheckIntervalSecondsdans` l'API Elastic Load Balancing
  - Intervalle sur la EC2 console Amazon
- Nombre de seuils sains : détermine le nombre de bilans de santé consécutifs requis avant de considérer qu'un contenant malsain est sain. Par défaut, l'équilibreur de charge nécessite cinq tests de santé réussis avant de signaler que le conteneur cible est sain.

  Ce paramètre est nommé :

  - `HealthyThresholdCountdans` l'API Elastic Load Balancing
  - Seuil sain sur la EC2 console Amazon

Important : pour les cibles nouvellement enregistrées, un seul bilan de santé réussi est nécessaire pour considérer que la cible est en bonne santé, quel que soit le seuil de santé défini. Le nombre de seuils sains ne s'applique que lorsqu'une cible passe d'un état malsain à un état sain.

Avec les paramètres par défaut, si une cible devient défectueuse puis se rétablit, le temps total nécessaire pour déterminer l'état d'un conteneur est de deux minutes et 30 secondes (`30 seconds * 5 = 150 seconds`).

Vous pouvez accélérer le processus de contrôle de santé si votre service démarre et se stabilise en moins de 10 secondes. Pour accélérer le processus, réduisez l'intervalle entre les bilans de santé et le nombre de seuils sains.

- `HealthCheckIntervalSeconds`(nom de l'API Elastic Load Balancing) ou Interval (nom de EC2 la console Amazon) : 5
- `HealthyThresholdCount`(nom de l'API Elastic Load Balancing) ou Healthy Threshold (nom de EC2 la console Amazon) : 2

Avec ce paramètre, le processus de vérification de l'état prend 10 secondes, contre deux minutes et 30 secondes par défaut.

Pour plus d'informations sur les paramètres de vérification de l'état d'Elastic Load Balancing, consultez la section Contrôles de santé de vos groupes cibles dans le guide de l'utilisateur d'Elastic Load Balancing.

Optimisez les paramètres de vidange des connexions de l'équilibreur de charge pour Amazon ECS

Pour permettre l'optimisation, les clients maintiennent une connexion permanente au service de conteneurs. Cela permet aux demandes ultérieures de ce client de réutiliser la connexion existante. Lorsque vous souhaitez arrêter le trafic vers un conteneur, vous devez en informer l'équilibreur de charge. L'équilibreur de charge vérifie régulièrement si le client a fermé la connexion Keep Alive. L'agent Amazon ECS surveille l'équilibreur de charge et attend que celui-ci indique que la connexion Keep Alive est fermée (la cible est dans un `UNUSED` état).

La durée pendant laquelle l'équilibreur de charge attend pour faire passer la cible à l'`UNUSED`état correspond au délai de désenregistrement. Vous pouvez configurer le paramètre d'équilibreur de charge suivant pour accélérer vos déploiements.

- `deregistration_delay.timeout_seconds`: 300 (par défaut)

Lorsque vous disposez d'un service dont le temps de réponse est inférieur à 1 seconde, définissez le paramètre sur la valeur suivante pour que l'équilibreur de charge n'attende que 5 secondes avant de rompre la connexion entre le client et le service principal :

- `deregistration_delay.timeout_seconds` : 5

> **Note**
>
> Ne définissez pas la valeur sur 5 secondes lorsqu'un service comporte des demandes de longue durée, telles que des téléchargements de fichiers lents ou des connexions de streaming.

Réactivité SIGTERM

Amazon ECS envoie d'abord un signal SIGTERM à la tâche pour indiquer que l'application doit se terminer et s'arrêter. Amazon ECS envoie ensuite un message SIGKILL. Lorsque les applications

ignorent le SIGTERM, le service Amazon ECS doit attendre avant d'envoyer le signal SIGKILL pour mettre fin au processus.

La durée pendant laquelle Amazon ECS attend avant d'envoyer le message SIGKILL est déterminée par l'option d'agent Amazon ECS suivante :

- `ECS_CONTAINER_STOP_TIMEOUT`: 30 (par défaut)

  Pour plus d'informations sur le paramètre de l'agent de conteneur, consultez Amazon ECS Container Agent on GitHub.

Pour accélérer le délai d'attente, définissez le paramètre de l'agent Amazon ECS sur la valeur suivante :

- `ECS_CONTAINER_STOP_TIMEOUT` : 2

  Si votre application prend plus d'une seconde, multipliez la valeur par 2 et utilisez ce nombre comme valeur.

Dans ce cas, Amazon ECS attend 2 secondes que le conteneur s'arrête, puis Amazon ECS envoie un message SIGKILL lorsque l'application ne s'arrête pas.

Vous pouvez également modifier le code de l'application pour intercepter le signal SIGTERM et y réagir. Voici un exemple dans JavaScript :

```
process.on('SIGTERM', function() {
  server.close();
})
```

Ce code oblige le serveur HTTP à arrêter d'écouter les nouvelles demandes, à terminer de répondre à toutes les demandes en cours, puis le processus Node.js s'arrête car la boucle d'événements n'a rien à voir. Cela étant, si le processus ne prend que 500 ms pour terminer ses demandes en cours de vol, il se termine plus tôt sans avoir à attendre la fin du délai d'arrêt et à recevoir un SIGKILL.

Utiliser un Application Load Balancer pour Amazon ECS

Un équilibreur de charge Application Load Balancer prend des décisions de routage au niveau de la couche d'application (HTTP/HTTPS), prend en charge le routage basé sur le chemin et peut acheminer les demandes vers un ou plusieurs ports de chaque instance de conteneur de votre

cluster. Les équilibreurs de charge Application Load Balancer prennent en charge le mappage de port hôte dynamique. Par exemple, si la définition de conteneur de la tâche spécifie le port 80 pour un port de conteneur NGINX et le port 0 pour le port hôte, le port hôte est choisi dynamiquement à partir de la plage de ports éphémères de l'instance de conteneur (par exemple, 32768 à 61000 sur la dernière AMI optimisée pour Amazon ECS). Lorsque la tâche est lancée, le conteneur NGINX est enregistré auprès de l'Application Load Balancer en tant que combinaison d'ID d'instance et de port, et le trafic est distribué à l'ID d'instance et au port correspondant à ce conteneur. Ce mappage dynamique vous permet d'exécuter plusieurs tâches à partir d'un seul service sur la même instance de conteneur. Pour plus d'informations, consultez le Guide de l'utilisateur pour les équilibreurs de charge Application Load Balancer.

Pour plus d'informations sur les meilleures pratiques en matière de définition de paramètres afin d'accélérer vos déploiements, consultez :

- Optimisation des paramètres de vérification de l'état de l'équilibreur de charge pour Amazon ECS
- Optimisez les paramètres de vidange des connexions de l'équilibreur de charge pour Amazon ECS

Tenez compte des points suivants lorsque vous utilisez des équilibreurs de charge d'application avec Amazon ECS :

- Amazon ECS nécessite le rôle IAM lié au service qui fournit les autorisations nécessaires pour enregistrer et désenregistrer des cibles dans votre équilibreur de charge lors de la création ou de l'arrêt de tâches. Pour de plus amples informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS.
- Le groupe cible doit avoir le type d'adresse IP défini sur IPv4.

  Bien que les tâches soient prises en charge IPv6, l'intégration des services avec IPv6 Elastic Load Balancing n'est pas prise en charge.
- Pour les services avec des tâches utilisant le mode réseau `awsvpc`, lorsque vous créez un groupe cible pour votre service, vous devez choisir `ip` comme type de cible, pas `instance`. Cela est dû au fait que les tâches qui utilisent le mode `awsvpc` réseau sont associées à une interface réseau élastique, et non à une EC2 instance Amazon.
- Si votre service nécessite l'accès à plusieurs ports d'équilibrage de charge, tels que le port 80 et le port 443 pour un HTTP/HTTPS service, vous pouvez configurer deux écouteurs. Un écouteur se charge du protocole HTTPS qui transmet la demande au service et un autre écouteur est responsable de la redirection des demandes HTTP vers le port HTTPS approprié. Pour plus

d'informations, consultez la section Création d'un écouteur pour votre Application Load Balancer dans le Guide de l'utilisateur pour les Application Load Balancers.

- La configuration de sous-réseau de votre équilibreur de charge doit inclure toutes les zones de disponibilité dans lesquels résident vos instances de conteneur.
- Une fois que vous avez créé un service, la configuration de l'équilibreur de charge ne peut pas être modifiée à partir de la AWS Management Console. Vous pouvez utiliser le AWS Copilot AWS CLI ou le SDK pour modifier la configuration de l'équilibreur de charge uniquement pour le contrôleur de déploiement `ECS` évolutif, et non AWS CodeDeploy pour le bleu/vert ou externe. AWS CloudFormation Lorsque vous ajoutez, mettez à jour ou supprimez une configuration d'équilibreur de charge, Amazon ECS lance un nouveau déploiement avec la configuration mise à jour d'Elastic Load Balancing. Cela entraîne l'enregistrement et le désenregistrement des tâches auprès des équilibreurs de charge. Nous vous recommandons de vérifier cela dans un environnement de test avant de mettre à jour la configuration d'Elastic Load Balancing. Pour plus d'informations sur la manière de modifier la configuration, consultez le UpdateServicemanuel Amazon Elastic Container Service API Reference.
- Si une tâche de service ne répond pas aux critères de vérification de l'état de l'équilibreur de charge, la tâche est arrêtée puis redémarrée. Ce processus continue jusqu'à ce que votre service atteigne le nombre souhaité de tâches en cours d'exécution.
- Si vous rencontrez des problèmes avec l'équilibreur de charge utilisé par vos services, consultez Résolution des problèmes liés aux équilibreurs de charge de service dans Amazon ECS.
- Lorsque vous utilisez le type de `instance` cible, vos tâches et votre équilibreur de charge doivent se trouver dans le même VPC. Lors de l'utilisation du type de `ip` cible, la connectivité inter-VPC est prise en charge.
- Utilisez un groupe cible unique pour chaque service.

  L'utilisation du même groupe cible pour plusieurs services peut entraîner des problèmes lors des déploiements de services.

Pour plus d'informations sur la création d'un équilibreur de charge d'application, voir Création d'un équilibreur de charge d'application dans les équilibreurs de charge d'application

Utiliser un Network Load Balancer pour Amazon ECS

Un Network Load Balancer prend des décisions de routage au niveau de la couche de transport (TCP/SSL). Il est capable de traiter des millions de requêtes par seconde. Lorsque l'équilibreur de charge reçoit une connexion, il sélectionne une cible depuis le groupe cible pour la règle par défaut

à l'aide d'un algorithme de routage du hachage de flux. Il tente d'ouvrir une connexion TCP à la cible sélectionnée sur le port spécifié dans la configuration de l'écouteur. Il transmet la demande sans modifier les en-têtes. Les équilibreurs de charge Network Load Balancer prennent en charge le mappage de port hôte dynamique. Par exemple, si la définition de conteneur de la tâche spécifie le port 80 pour un port de conteneur NGINX et le port 0 pour le port hôte, le port hôte est choisi dynamiquement à partir de la plage de ports éphémères de l'instance de conteneur (par exemple, 32768 à 61000 sur la dernière AMI optimisée pour Amazon ECS). Lorsque la tâche est lancée, le conteneur NGINX est enregistré dans l'équilibreur de charge Network Load Balancer en tant que combinaison d'ID d'instance et de port, et le trafic est réparti vers l'ID d'instance et le port correspondant à ce conteneur. Ce mappage dynamique vous permet d'exécuter plusieurs tâches à partir d'un seul service sur la même instance de conteneur. Pour plus d'informations, veuillez consulter le Guide de l'utilisateur pour les Network Load Balancers.

Pour plus d'informations sur les meilleures pratiques en matière de définition de paramètres afin d'accélérer vos déploiements, consultez :

- Optimisation des paramètres de vérification de l'état de l'équilibreur de charge pour Amazon ECS
- Optimisez les paramètres de vidange des connexions de l'équilibreur de charge pour Amazon ECS

Tenez compte des points suivants lorsque vous utilisez des équilibreurs de charge réseau avec Amazon ECS :

- Amazon ECS nécessite le rôle IAM lié au service qui fournit les autorisations nécessaires pour enregistrer et désenregistrer des cibles dans votre équilibreur de charge lors de la création ou de l'arrêt de tâches. Pour de plus amples informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS.
- Vous ne pouvez pas associer plus de cinq groupes cibles à un service.
- Pour les services avec des tâches utilisant le mode réseau `awsvpc`, lorsque vous créez un groupe cible pour votre service, vous devez choisir `ip` comme type de cible, pas `instance`. Cela est dû au fait que les tâches qui utilisent le mode `awsvpc` réseau sont associées à une interface réseau élastique, et non à une EC2 instance Amazon.
- La configuration de sous-réseau de votre équilibreur de charge doit inclure toutes les zones de disponibilité dans lesquels résident vos instances de conteneur.
- Une fois que vous avez créé un service, la configuration de l'équilibreur de charge ne peut pas être modifiée à partir de la AWS Management Console. Vous pouvez utiliser le AWS Copilot AWS CLI ou le SDK pour modifier la configuration de l'équilibreur de charge uniquement pour le

contrôleur de déploiement `ECS` évolutif, et non AWS CodeDeploy pour le bleu/vert ou externe. AWS CloudFormation Lorsque vous ajoutez, mettez à jour ou supprimez une configuration d'équilibreur de charge, Amazon ECS lance un nouveau déploiement avec la configuration mise à jour d'Elastic Load Balancing. Cela entraîne l'enregistrement et le désenregistrement des tâches auprès des équilibreurs de charge. Nous vous recommandons de vérifier cela dans un environnement de test avant de mettre à jour la configuration d'Elastic Load Balancing. Pour plus d'informations sur la manière de modifier la configuration, consultez le UpdateServicemanuel Amazon Elastic Container Service API Reference.

- Si une tâche de service ne répond pas aux critères de vérification de l'état de l'équilibreur de charge, la tâche est arrêtée puis redémarrée. Ce processus continue jusqu'à ce que votre service atteigne le nombre souhaité de tâches en cours d'exécution.
- Lorsque vous utilisez un Gateway Load Balancer configuré avec des adresses IP comme cibles et la préservation de l'adresse IP du client désactivée, les demandes sont considérées comme provenant de l'adresse IP privée du Gateway Load Balancer. Cela signifie que les services associés à un Gateway Load Balancer sont effectivement ouverts au monde dès que vous autorisez les demandes entrantes et les contrôles de santé dans le groupe de sécurité cible.
- Pour les tâches Fargate, vous devez utiliser la `1.4.0` version de plate-forme (Linux) `1.0.0` ou (Windows).
- Si vous rencontrez des problèmes avec l'équilibreur de charge utilisé par vos services, consultez Résolution des problèmes liés aux équilibreurs de charge de service dans Amazon ECS.
- Lorsque vous utilisez le type de `instance` cible, vos tâches et votre équilibreur de charge doivent se trouver dans le même VPC. Lors de l'utilisation du type de `ip` cible, la connectivité inter-VPC est prise en charge.
- La préservation de l'adresse IP du client Network Load Balancer est compatible avec les cibles Fargate.
- Utilisez un groupe cible unique pour chaque service.

  L'utilisation du même groupe cible pour plusieurs services peut entraîner des problèmes lors des déploiements de services.

Pour plus d'informations sur la création d'un Network Load Balancer, voir Create a Network Load Balancer dans Network Load Balancers

> **Important**
>
> Si la définition de tâche de votre service utilise le mode réseau `awsvpc` (requis pour le type de lancement Fargate), vous devez choisir le type de cible `ip`, et non `instance`. Cela est dû au fait que les tâches qui utilisent le mode `awsvpc` réseau sont associées à une interface réseau élastique, et non à une EC2 instance Amazon.
> Vous ne pouvez pas enregistrer des instances par ID d'instance si elles ont les types d'instance suivants : C1 CC1 CC2, CG1, CG2, CR1,, G1, G2, HI1, M1 HS1, M2, M3 et T1. Vous pouvez enregistrer les instances de ces types par adresse IP.

Utiliser un Gateway Load Balancer pour Amazon ECS

Les équilibreurs Gateway Load Balancer agissent au niveau de la troisième couche du modèle OSI Open Systems Interconnection (OSI), la couche réseau. Ils écoutent tous les paquets IP sur tous les ports et transfèrent le trafic vers le groupe cible spécifié dans la règle d'écoute. Il maintient la fidélité des flux à une appliance cible spécifique en utilisant 5 tuples (pour les TCP/UDP flux) ou 3 tuples (pour les flux non TCP/UDP). Par exemple, si la définition de conteneur de la tâche spécifie le port 80 pour un port de conteneur NGINX et le port 0 pour le port hôte, le port hôte est choisi dynamiquement à partir de la plage de ports éphémères de l'instance de conteneur (par exemple, 32768 à 61000 sur la dernière AMI optimisée pour Amazon ECS). Lorsque la tâche est lancée, le conteneur NGINX est enregistré auprès du Gateway Load Balancer en tant que combinaison d'ID d'instance et de port, et le trafic est distribué à l'ID d'instance et au port correspondant à ce conteneur. Ce mappage dynamique vous permet d'exécuter plusieurs tâches à partir d'un seul service sur la même instance de conteneur. Pour plus d'informations, consultez Qu'est-ce qu'un Gateway Load Balancer dans Gateway Load Balancers.

Pour plus d'informations sur les meilleures pratiques en matière de définition de paramètres afin d'accélérer vos déploiements, consultez :

- Optimisation des paramètres de vérification de l'état de l'équilibreur de charge pour Amazon ECS
- Optimisez les paramètres de vidange des connexions de l'équilibreur de charge pour Amazon ECS

Tenez compte des points suivants lorsque vous utilisez des équilibreurs de charge de passerelle avec Amazon ECS :

- Amazon ECS nécessite le rôle IAM lié au service qui fournit les autorisations nécessaires pour enregistrer et désenregistrer des cibles dans votre équilibreur de charge lors de la création ou de l'arrêt de tâches. Pour de plus amples informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS.
- Pour les services avec des tâches utilisant le mode réseau `awsvpc`, lorsque vous créez un groupe cible pour votre service, vous devez choisir `ip` comme type de cible, pas `instance`. Cela est dû au fait que les tâches qui utilisent le mode `awsvpc` réseau sont associées à une interface réseau élastique, et non à une EC2 instance Amazon.
- Pour les services dont les tâches utilisent un mode réseau autre que `awsvpc` les équilibreurs de charge de passerelle ne sont pas pris en charge.
- La configuration de sous-réseau de votre équilibreur de charge doit inclure toutes les zones de disponibilité dans lesquels résident vos instances de conteneur.
- Une fois que vous avez créé un service, la configuration de l'équilibreur de charge ne peut pas être modifiée à partir de la AWS Management Console. Vous pouvez utiliser le AWS Copilot AWS CLI ou le SDK pour modifier la configuration de l'équilibreur de charge uniquement pour le contrôleur de déploiement `ECS` évolutif, et non AWS CodeDeploy pour le bleu/vert ou externe. AWS CloudFormation Lorsque vous ajoutez, mettez à jour ou supprimez une configuration d'équilibreur de charge, Amazon ECS lance un nouveau déploiement avec la configuration mise à jour d'Elastic Load Balancing. Cela entraîne l'enregistrement et le désenregistrement des tâches auprès des équilibreurs de charge. Nous vous recommandons de vérifier cela dans un environnement de test avant de mettre à jour la configuration d'Elastic Load Balancing. Pour plus d'informations sur la manière de modifier la configuration, consultez le UpdateServicemanuel Amazon Elastic Container Service API Reference.
- Si une tâche de service ne répond pas aux critères de vérification de l'état de l'équilibreur de charge, la tâche est arrêtée puis redémarrée. Ce processus continue jusqu'à ce que votre service atteigne le nombre souhaité de tâches en cours d'exécution.
- Lorsque vous utilisez un Gateway Load Balancer configuré avec des adresses IP comme cibles, les demandes sont considérées comme provenant de l'adresse IP privée du Gateway Load Balancer. Cela signifie que les services associés à un Gateway Load Balancer sont effectivement ouverts au monde dès que vous autorisez les demandes entrantes et les contrôles de santé dans le groupe de sécurité cible.
- Pour les tâches Fargate, vous devez utiliser la `1.4.0` version de plate-forme (Linux) `1.0.0` ou (Windows).
- Si vous rencontrez des problèmes avec l'équilibreur de charge utilisé par vos services, consultez Résolution des problèmes liés aux équilibreurs de charge de service dans Amazon ECS.

- Lorsque vous utilisez le type de `instance` cible, vos tâches et votre équilibreur de charge doivent se trouver dans le même VPC. Lors de l'utilisation du type de `ip` cible, la connectivité inter-VPC est prise en charge.
- Utilisez un groupe cible unique pour chaque service.

  L'utilisation du même groupe cible pour plusieurs services peut entraîner des problèmes lors des déploiements de services.

Pour plus d'informations sur la création d'un Gateway Load Balancer, voir Getting started with Gateway Load Balancers in Gateway Load Balancers

> **Important**
>
> Si la définition de tâche de votre service utilise le mode réseau `awsvpc` (requis pour le type de lancement Fargate), vous devez choisir le type de cible `ip`, et non `instance`. Cela est dû au fait que les tâches qui utilisent le mode `awsvpc` réseau sont associées à une interface réseau élastique, et non à une EC2 instance Amazon.
> Vous ne pouvez pas enregistrer des instances par ID d'instance si elles ont les types d'instance suivants : C1 CC1 CC2, CG1, CG2, CR1,, G1, G2, HI1, M1 HS1, M2, M3 et T1. Vous pouvez enregistrer les instances de ces types par adresse IP.

Enregistrement de plusieurs groupes cibles auprès d'un service Amazon ECS

Votre service Amazon ECS service peut desservir le trafic provenant de plusieurs équilibreurs de charge et exposer plusieurs ports à charge équilibrée en spécifiant plusieurs groupes cibles dans une définition de service.

Pour créer un service spécifiant plusieurs groupes cibles, vous devez créer le service à l'aide de l'API Amazon ECS, du SDK ou d'un AWS CloudFormation modèle. AWS CLI Une fois que le service est créé, vous pouvez afficher le service et les groupes cibles enregistrés auprès de celui-ci avec la AWS Management Console. Vous devez utiliser `UpdateService` pour modifier la configuration de l'équilibreur de charge d'un service existant.

Plusieurs groupes cibles peuvent être spécifiés dans une définition de service à l'aide du format suivant. Pour obtenir la syntaxe complète d'une définition de service, consultez Modèle de définition de service.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"container_name",
      "containerPort":container_port
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"container_name",
      "containerPort":container_port
   }
]
```

Considérations

Les informations suivantes doivent être prises en compte lorsque vous spécifiez plusieurs groupes cibles dans une définition de service.

- Pour les services qui utilisent un équilibreur Application Load Balancer ou un Network Load Balancer, vous ne pouvez pas attacher plus de cinq groupes cibles à un service.
- La spécification de plusieurs groupes cibles dans une définition de service n'est prise en charge que dans les conditions suivantes :
  - Le service doit utiliser un équilibreur de charge Application Load Balancer ou un Network Load Balancer.
  - Le service doit utiliser la mise à jour propagée (`ECS`) comme type de contrôleur de déploiement.
- La spécification de plusieurs groupes cibles est prise en charge pour les services contenant des tâches utilisant à la fois les types Fargate et Launch. EC2
- Lors de la création d'un service qui spécifie plusieurs groupes cibles, le rôle lié à un service Amazon ECS service doit être créé. Le rôle est créé en omettant le paramètre `role` dans les demandes d'API, ou la propriété `Role` dans AWS CloudFormation. Pour de plus amples informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS.

Exemples de définitions de service

Voici quelques exemples de cas d'utilisation pour spécifier plusieurs groupes cibles dans une définition de service. Pour obtenir la syntaxe complète d'une définition de service, consultez Modèle de définition de service.

Disposer d'équilibreurs de charge séparés pour le trafic interne et externe

Dans le cas d'utilisation suivant, un service utilise deux équilibreurs de charge distincts, l'un pour le trafic interne et l'autre pour le trafic Internet, pour le même conteneur et port.

```
"loadBalancers":[
   //Internal ELB
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"nginx",
      "containerPort":8080
   },
   //Internet-facing ELB
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"nginx",
      "containerPort":8080
   }
]
```

Exposer plusieurs ports depuis le même conteneur

Dans le cas d'utilisation suivant, un service utilise un équilibreur de charge, mais expose plusieurs ports à partir du même conteneur. Par exemple, un conteneur Jenkins expose le port 8080 pour l'interface web Jenkins et le port 50000 pour l'API.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"jenkins",
      "containerPort":8080
```

```
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"jenkins",
      "containerPort":50000
   }
]
```

Exposer les ports de plusieurs conteneurs

Dans le cas d'utilisation suivant, un service utilise un équilibreur de charge et deux groupes cibles pour exposer les ports de conteneurs distincts.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"webserver",
      "containerPort":80
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"database",
      "containerPort":3306
   }
]
```

Faites évoluer automatiquement votre service Amazon ECS

Le dimensionnement automatique permet d'augmenter ou de diminuer automatiquement le nombre de tâches souhaité dans votre service Amazon ECS. Amazon ECS utilise le service Application Auto Scaling pour fournir cette fonctionnalité. Pour de plus amples informations, veuillez consulter le Guide de l'utilisateur Application Auto Scaling.

Amazon ECS publie des CloudWatch statistiques indiquant l'utilisation moyenne du processeur et de la mémoire de votre service. Pour de plus amples informations, veuillez consulter Mesures d'utilisation des services Amazon ECS. Vous pouvez utiliser ces CloudWatch indicateurs, ainsi que

d'autres, pour étendre votre service (ajouter des tâches) afin de répondre à la forte demande aux heures de pointe, et pour étendre votre service (exécuter moins de tâches) afin de réduire les coûts pendant les périodes de faible utilisation.

Amazon ECS Service Auto Scaling prend en charge les types de scalabilité automatique suivants :

- Utiliser une métrique cible pour faire évoluer les services Amazon ECS – Augmente ou réduit le nombre de tâches exécutées par votre service en fonction d'une valeur cible pour une métrique spécifique. Cette option est similaire à la façon dont votre thermostat maintient la température de votre domicile. Vous sélectionnez une température et le thermostat se charge du reste.
- Utilisez des incréments prédéfinis basés sur les CloudWatch alarmes pour dimensionner les services Amazon ECS – Augmente ou réduit le nombre de tâches exécutées par votre service en fonction d'un ensemble d'ajustements de mise à l'échelle, appelés ajustements d'étape, qui varient en fonction de la valeur du seuil de l'alarme.
- Utilisez des actions planifiées pour faire évoluer les services Amazon ECS—Augmentez ou diminuez le nombre de tâches exécutées par votre service en fonction de la date et de l'heure.
- Utilisez des modèles historiques pour faire évoluer les services Amazon ECS grâce à une mise à l'échelle prédictive—Augmentez ou diminuez le nombre de tâches exécutées par votre service en fonction de l'analyse des données de charge historiques afin de détecter les tendances quotidiennes ou hebdomadaires des flux de trafic.

Considérations

Lorsque vous utilisez des stratégies de mise à l'échelle, tenez compte des informations suivantes :

- Amazon ECS envoie des métriques toutes les 1 minute à CloudWatch. Les métriques ne sont pas disponibles tant que les clusters et les services ne les ont pas envoyées CloudWatch, et vous ne pouvez pas créer d' CloudWatch alarmes pour des métriques qui n'existent pas.
- Les stratégies de mise à l'échelle prennent en charge un temps de stabilisation. Il s'agit de la durée, en secondes, à attendre qu'une activité de mise à l'échelle précédente prenne effet.
  - Pour les événements de montée en puissance, l'intention est de réduire continuellement (mais pas excessivement) la montée en puissance. Une fois que Service Auto Scaling a réussi une montée en puissance à l'aide d'une stratégie de mise à l'échelle, l'application commence à calculer le temps de stabilisation. La politique de mise à l'échelle n'augmente pas à nouveau la capacité souhaitée, sauf si une plus grande montée en puissance est lancée ou si le temps de stabilisation est écoulé. Tandis que le temps de stabilisation de la montée en puissance

s'applique, la capacité ajoutée par l'activité de mise à l'échelle initiale est calculée dans le cadre de la capacité souhaitée pour la prochaine activité de montée en puissance.

- Pour les événements de mise à l'échelle horizontale, l'objectif est de procéder à une mise à l'échelle prudente afin de protéger la disponibilité de votre application, de sorte que les activités de mise à l'échelle horizontale sont bloquées jusqu'à l'expiration du temps de stabilisation. Toutefois, si une autre alarme lance une activité de montée en puissance au cours du temps de stabilisation de la diminution de charge, Application Auto Scaling monte immédiatement en puissance la cible. Dans ce cas, le temps de stabilisation de la mise à l'échelle horizontale s'arrête et ne se termine pas.

- Le planificateur de service respecte le nombre souhaité à tout moment mais, tant que vous avez des stratégies de mise à l'échelle et des alarmes actives sur un service, Service Auto Scaling peut modifier un nombre souhaité manuellement défini par vous-même.
- Si le nombre souhaité d'un service est défini en dessous de sa valeur de capacité minimale et qu'une alarme déclenche une activité de scale-out, Service Auto Scaling redimensionne le nombre souhaité jusqu'à la valeur de capacité minimale, puis continue à évoluer selon les besoins, en fonction de la politique de dimensionnement associée à l'alarme. Toutefois, une activité de mise à l'échelle horizontale ne modifie pas le nombre souhaité, car il est déjà inférieur à la valeur de capacité minimale.
- Si le nombre souhaité d'un service est défini au-dessus de sa valeur de capacité maximale et qu'une alarme déclenche une échelle d'activité, Service Auto Scaling redimensionne le décompte souhaité jusqu'à la valeur de capacité maximale, puis continue à évoluer selon les besoins, en fonction de la politique de dimensionnement associée à l'alarme. Toutefois, une activité de montée en puissance ne modifie pas le nombre souhaité, car il est déjà supérieur à la valeur de capacité maximale.
- Au cours des activités de mise à l'échelle, le nombre réel de tâches en cours d'exécution dans un service correspond à la valeur utilisée par Service Auto Scaling comme point de départ, plutôt qu'au nombre souhaité. C'est ce qui est supposé représenter la capacité de traitement. Cela évite une mise à l'échelle excessive qui ne pourrait pas être satisfaite, lorsqu'il n'y a par exemple pas assez de ressources d'instances de conteneur pour placer les tâches supplémentaires. Si la capacité d'instance de conteneur est disponible ultérieurement, l'activité de mise à l'échelle en suspens peut réussir et d'autres activités de mise à l'échelle peuvent continuer après la période de stabilisation.
- Si vous souhaitez que votre nombre de tâches soit réduit à zéro lorsqu'il n'y a pas de travail à effectuer, définissez une capacité minimale de 0. Avec les stratégies de suivi des objectifs et d'échelonnement, lorsque la capacité réelle est de 0 et que la métrique indique qu'il y a une

requête d'application, Service Auto Scaling attend l'envoi d'un point de données avant d'effectuer la montée en puissance. Dans ce cas, la mise à l'échelle entraîne la quantité la plus petite possible comme point de départ, puis reprend la mise à l'échelle en fonction du nombre réel de tâches en cours d'exécution.

- Application Auto Scaling désactive les processus évolutifs pendant que les déploiements Amazon ECS sont en cours. Toutefois, pendant les déploiements, les processus de montée en puissance se poursuivent, sauf s'ils sont suspendus. Ce comportement ne s'applique pas aux services Amazon ECS utilisant le contrôleur de déploiement externe. Pour de plus amples informations, veuillez consulter Service Auto Scaling et déploiements.
- Vous disposez de plusieurs options d'Application Auto Scaling pour les tâches Amazon ECS. Le suivi des cibles est le mode le plus simple à utiliser. Il vous suffit de définir une valeur cible pour une métrique, telle que l'utilisation moyenne du processeur. Ensuite, l'autoscaler gère automatiquement le nombre de tâches nécessaires pour atteindre cette valeur. Grâce à la mise à l'échelle par étapes, vous pouvez réagir plus rapidement aux variations de la demande, car vous définissez les seuils spécifiques pour vos métriques de mise à l'échelle et le nombre de tâches à ajouter ou à supprimer lorsque les seuils sont dépassés. Et surtout, vous pouvez réagir très rapidement aux variations de la demande en réduisant la durée pendant laquelle une alarme de seuil est franchie.

Pour plus d'informations sur les meilleures pratiques en matière de mise à l'échelle automatique des services, consultezOptimisation de la mise à l'échelle automatique du service Amazon ECS.

Service Auto Scaling et déploiements

Application Auto Scaling désactive les processus évolutifs pendant que les déploiements Amazon ECS sont en cours. Toutefois, pendant les déploiements, les processus de montée en puissance se poursuivent, sauf s'ils sont suspendus. Ce comportement ne s'applique pas aux services Amazon ECS utilisant le contrôleur de déploiement externe. Si vous souhaitez suspendre les processus de montée en puissance pendant les déploiements, procédez comme suit.

1. Appelez la describe-scalable-targetscommande en spécifiant l'ID de ressource du service associé à la cible évolutive dans Application Auto Scaling (Example :`service/default/sample-webapp`). Enregistrez la sortie. Vous en aurez besoin pour appeler la commande suivante.
2. Appelez la register-scalable-targetcommande en spécifiant l'ID de ressource, l'espace de noms et la dimension évolutive. Spécifiez `true` pour `DynamicScalingInSuspended` et `DynamicScalingOutSuspended`.

3. Une fois le déploiement terminé, vous pouvez appeler la register-scalable-targetcommande pour reprendre le dimensionnement.

Pour de plus amples informations, veuillez consulter Suspension et reprise de la mise à l'échelle pour Application Auto Scaling.

Utiliser une métrique cible pour faire évoluer les services Amazon ECS

Grâce aux politiques de suivi des objectifs et d'échelonnement, vous sélectionnez une métrique et définissez une valeur cible. Amazon ECS Service Auto Scaling crée et gère les CloudWatch alarmes qui contrôlent la politique de dimensionnement et calcule l'ajustement de dimensionnement en fonction de la métrique et de la valeur cible. La politique de mise à l'échelle ajoute ou supprime des tâches de service si nécessaire pour maintenir la métrique à la valeur cible spécifiée ou proche de celle-ci. En plus de maintenir la métrique proche de la valeur cible, une politique de suivi des objectifs et d'échelonnement s'ajuste également aux fluctuations de la métrique dues à un modèle de charge fluctuant, et minimise les fluctuations rapides du nombre de tâches exécutées dans votre service.

Les politiques de suivi des cibles éliminent le besoin de définir manuellement les CloudWatch alarmes et les ajustements de dimensionnement. Amazon ECS gère cela automatiquement en fonction de l'objectif que vous avez défini.

Tenez compte des éléments suivants lorsque vous utilisez des politiques de suivi des cibles :

- Une politique de suivi des objectifs et d'échelonnement suppose qu'elle doit effectuer une montée en puissance ; lorsque la métrique spécifiée est au-dessus de la valeur cible. Vous ne pouvez pas utiliser une politique de suivi des objectifs et d'échelonnement pour effectuer une montée en puissance lorsque la métrique spécifiée est en dessous de la valeur cible.
- Une politique de suivi des objectifs et d'échelonnement n'effectue pas de mise à l'échelle lorsque la métrique spécifiée a des données insuffisantes. Elle n'effectue pas de mise à l'échelle horizontale car elle n'interprète pas des données insuffisantes comme une faible utilisation.
- Vous pouvez constater des écarts entre la valeur cible et les points de données de métrique réels. Ceci est dû au fait que Service Auto Scaling agit toujours avec prudence en effectuant un arrondi vers le haut ou vers le bas quand il détermine la capacité à ajouter ou à enlever. Cela l'empêche d'ajouter une capacité insuffisante ou de retirer trop de capacité.
- Pour garantir la disponibilité de l'application, le service augmente proportionnellement aux métriques aussi rapidement que possible, mais diminue plus progressivement.

- Application Auto Scaling désactive les processus évolutifs pendant que les déploiements Amazon ECS sont en cours. Toutefois, pendant les déploiements, les processus de montée en puissance se poursuivent, sauf s'ils sont suspendus. Ce comportement ne s'applique pas aux services Amazon ECS utilisant le contrôleur de déploiement externe. Pour de plus amples informations, veuillez consulter Service Auto Scaling et déploiements.
- Vous pouvez avoir plusieurs politiques de suivi des objectifs et d'échelonnement pour un service Amazon ECS service dans la mesure où chacune d'elles utilise une métrique différente. L'objectif de Service Auto Scaling est de toujours donner la priorité à la disponibilité, afin que son comportement diffère selon que les politiques de suivi des objectifs et d'échelonnement sont prêtes pour une augmentation ou une diminution de taille. Il procèdera à la montée en puissance du service si l'une des politiques Suivi de la cible est prête pour une augmentation de taille, mais la diminuera uniquement si toutes les politiques Suivi de la cible (avec la portion de mise à l'échelle horizontale activée) sont prêtes pour une diminution de taille.
- Ne modifiez ni ne supprimez les CloudWatch alarmes gérées par Service Auto Scaling dans le cadre d'une politique de dimensionnement du suivi des cibles. Service Auto Scaling supprime les alarmes automatiquement quand vous supprimez la politique de mise à l'échelle.
- La `ALBRequestCountPerTarget` métrique pour les politiques de dimensionnement du suivi des cibles n'est pas prise en charge pour le type de blue/green déploiement.

Pour de plus amples informations sur les politiques de dimensionnement de suivi de cible, veuillez consulter Politiques de dimensionnement de suivi de cible dans le Manuel de l'utilisateur Application Auto Scaling.

Création d'une politique de dimensionnement du suivi des cibles pour le dimensionnement automatique du service Amazon ECS

Créez une politique de dimensionnement du suivi des cibles pour qu'Amazon ECS augmente ou diminue automatiquement le nombre de tâches souhaité dans votre service. Le suivi des cibles fonctionne à partir d'une valeur métrique cible.

console

1. Outre les autorisations IAM standard pour créer et mettre à jour des services, vous avez besoin d'autorisations supplémentaires. Pour de plus amples informations, veuillez consulter Autorisations IAM requises pour le dimensionnement automatique du service Amazon ECS.
2. Déterminez les mesures à utiliser pour la politique. Les mesures suivantes sont disponibles :

- ECSServiceMoyenne CPUUtilization : utilisation moyenne du processeur que le service doit utiliser.
- ECSServiceAverageMemoryUtilization— Utilisation moyenne de la mémoire que le service doit utiliser.
- ALBRequestCountPerTarget— Le nombre moyen de demandes par minute que cette tâche devrait idéalement recevoir.

3. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
4. Sur la page Clusters, choisissez le cluster.
5. Sur la page des détails du cluster, dans la section Services, puis choisissez le service.

   La page des détails du service apparaît.
6. Choisissez Définir le nombre de tâches.
7. Sous Nombre de tâches de service Amazon ECS, sélectionnez Utiliser le dimensionnement automatique.

   La section Nombre de tâches apparaît.

   a. Dans le champ Nombre minimum de tâches, entrez la limite inférieure du nombre de tâches à utiliser pour le dimensionnement automatique du service. Le nombre souhaité ne sera pas inférieur à ce nombre.
   b. Pour Maximum, entrez la limite supérieure du nombre de tâches à utiliser par Service Auto Scaling. Le nombre souhaité ne sera pas supérieur à ce nombre.
   c. Choisissez Enregistrer.

      La page des politiques apparaît.
8. Choisissez Créer une politique de dimensionnement.

   La page Créer une politique apparaît.
9. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Target tracking (Suivi de cible).
10. Pour Policy name (Nom de la politique), saisissez un nom de politique.
11. Pour le type de métrique, choisissez vos métriques dans la liste des options.
12. Pour Utilisation cible, entrez la valeur cible pour le pourcentage de tâches qu'Amazon ECS doit gérer. Le dimensionnement automatique du service augmente votre capacité jusqu'à ce que

l'utilisation moyenne atteigne le taux d'utilisation cible ou jusqu'à ce qu'elle atteigne le nombre maximum de tâches que vous avez spécifié.

13. Sous Paramètres supplémentaires, procédez comme suit

    a. Pour la période de recharge progressive, entrez le délai en secondes qui s'écoule entre la fin d'une activité d'extension et le début d'une autre activité de mise à l'échelle.

    b. Pour la période de recharge de mise à l'échelle progressive, entrez le délai en secondes pendant lequel vous devez attendre qu'une activité de mise en échelle progressive précédente prenne effet.

    c. Pour créer uniquement une politique de scale-out, sélectionnez Désactiver le scale-in.

14. Choisissez Créer une politique de dimensionnement.

AWS CLI

1. Enregistrez votre service Amazon ECS en tant que cible évolutive à l'aide de la register-scalable-targetcommande.
2. Créez une politique de dimensionnement à l'aide de la put-scaling-policycommande.

Utilisez des incréments prédéfinis basés sur les CloudWatch alarmes pour dimensionner les services Amazon ECS

Les politiques de dimensionnement par étapes vous permettent de créer et de gérer les CloudWatch alarmes qui déclenchent le processus de dimensionnement. Lorsqu'une alarme est violée, Amazon ECS initie la politique de dimensionnement associée à cette alarme. La politique d'échelonnement des étapes permet d'échelonner les tâches à l'aide d'un ensemble d'ajustements, appelés ajustements par étapes. La taille de l’ajustement varie en fonction de l’ampleur du déclenchement de l’alarme.

- Si la violation dépasse le premier seuil, Amazon ECS applique le premier ajustement.
- Si la violation dépasse le deuxième seuil, Amazon ECS applique le deuxième ajustement, et ainsi de suite.

Nous vous recommandons vivement d'utiliser des politiques de dimensionnement du suivi des cibles pour effectuer une mise à l'échelle en fonction de mesures telles que l'utilisation moyenne du processeur ou le nombre moyen de demandes par cible. Les indicateurs qui diminuent lorsque la

capacité augmente et augmentent lorsque la capacité diminue peuvent être utilisés pour augmenter proportionnellement le nombre de tâches à l'aide du suivi des cibles. Cela permet de garantir qu'Amazon ECS suit de près la courbe de demande pour vos applications.

Création d'une politique de dimensionnement par étapes pour le dimensionnement automatique du service Amazon ECS

Créez une politique d'échelonnement pour qu'Amazon ECS augmente ou diminue automatiquement le nombre de tâches souhaité dans votre service. Le dimensionnement par étapes est basé sur un ensemble d'ajustements de mise à l'échelle, appelés ajustements par étapes, qui varient en fonction de l'ampleur de la brèche d'alarme.

console

1. Outre les autorisations IAM standard pour créer et mettre à jour des services, vous avez besoin d'autorisations supplémentaires. Pour de plus amples informations, veuillez consulter Autorisations IAM requises pour le dimensionnement automatique du service Amazon ECS.
2. Déterminez les mesures à utiliser pour la politique. Les mesures suivantes sont disponibles :
   - ECSServiceMoyenne CPUUtilization : utilisation moyenne du processeur que le service doit utiliser.
   - ECSServiceAverageMemoryUtilization— Utilisation moyenne de la mémoire que le service doit utiliser.
   - ALBRequestCountPerTarget— Le nombre moyen de demandes par minute que cette tâche devrait idéalement recevoir.
3. Créez les CloudWatch alarmes pour les métriques. Pour plus d'informations, consultez la section Création CloudWatch d'une alarme basée sur un seuil statique dans le guide de CloudWatch l'utilisateur Amazon.
4. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
5. Sur la page Clusters, choisissez le cluster.
6. Sur la page des détails du cluster, dans la section Services, puis choisissez le service.

   La page des détails du service apparaît.
7. Choisissez Définir le nombre de tâches.
8. Sous Nombre de tâches de service Amazon ECS, sélectionnez Utiliser le dimensionnement automatique.

La section Nombre de tâches apparaît.

a. Dans le champ Nombre minimum de tâches, entrez la limite inférieure du nombre de tâches à utiliser pour le dimensionnement automatique du service. Le nombre souhaité ne sera pas inférieur à ce nombre.

b. Pour Maximum, entrez la limite supérieure du nombre de tâches à utiliser par Service Auto Scaling. Le nombre souhaité ne sera pas supérieur à ce nombre.

c. Choisissez Enregistrer.

   La page des politiques apparaît.

9. Choisissez Créer une politique de dimensionnement.

   La page Créer une politique apparaît.

10. Pour le type de politique de dimensionnement, choisissez Step Scaling.

11. Configurez les propriétés de scaling-out. Sous Étapes pour ajouter des tâches, procédez comme suit :

    a. Pour Policy name (Nom de la politique), saisissez un nom de politique.

    b. Pour le nom de l'CloudWatch alarme, choisissez l' CloudWatch alarme.

    c. Pour le type d'agrégation de mesures, choisissez comment comparer la métrique sélectionnée au seuil défini.

    d. Pour les types d'ajustement A, choisissez si l'ajustement est basé sur une modification du nombre de tâches ou sur une modification du pourcentage de tâches.

    e. Dans Actions à entreprendre, entrez les valeurs correspondant à l'action à entreprendre.

       Choisissez Ajouter une étape pour ajouter des actions supplémentaires.

12. Configurez les propriétés de mise à l'échelle. Sous Étapes pour supprimer des tâches, procédez comme suit :

    a. Pour Policy name (Nom de la politique), saisissez un nom de politique.

    b. Pour le nom de l'CloudWatch alarme, choisissez l' CloudWatchalarme.

    c. Pour le type d'agrégation de mesures, choisissez comment comparer la métrique sélectionnée au seuil défini.

    d. Pour les types d'ajustement, choisissez si l'ajustement est basé sur une modification du nombre de tâches ou sur une modification du pourcentage de tâches.

e. Dans Actions à entreprendre, entrez les valeurs correspondant à l'action à entreprendre.

   Choisissez Ajouter une étape pour ajouter des actions supplémentaires.

13. Pour la période de recharge, entrez le temps, en secondes, nécessaire pour qu'une activité de dimensionnement précédente prenne effet. Dans le cas d'une politique d'ajout, il s'agit du moment où, après une activité de scale-out, la politique de scale-out bloque les activités de scale-in et limite le nombre de tâches pouvant être redimensionnées à la fois. Pour une politique de suppression, il s'agit de la période qui suit une activité d'extension qui doit se terminer avant qu'une autre activité d'extension ne puisse démarrer.
14. Choisissez Créer une politique de dimensionnement.

AWS CLI

1. Enregistrez votre service Amazon ECS en tant que cible évolutive à l'aide de la register-scalable-targetcommande.
2. Créez une politique de dimensionnement à l'aide de la put-scaling-policycommande.

Utilisez des actions planifiées pour faire évoluer les services Amazon ECS

Avec le dimensionnement planifié, vous pouvez configurer le dimensionnement automatique de votre application en fonction de changements de charge prévisibles en créant des actions planifiées qui augmentent ou diminuent le nombre de tâches à des moments précis. Cette procédure vous permet de mettre votre application à l’échelle de manière proactive afin qu’elle corresponde aux variations de charge prévisibles.

Ces actions de mise à l’échelle planifiées vous permettent d’optimiser les coûts et les performances. Votre application dispose d'un nombre de tâches suffisant pour faire face au pic de trafic en milieu de semaine, mais elle n'en augmente pas le nombre à d'autres moments.

Vous pouvez utiliser conjointement la mise à l’échelle planifiée et les stratégies de mise à l’échelle pour bénéficier des avantages des approches proactives et réactives de la mise à l’échelle. Après l'exécution d'une action de dimensionnement planifiée, la politique de dimensionnement peut continuer à prendre des décisions quant à l'opportunité d'augmenter davantage le nombre de tâches. Cela vous permet de vous assurer que vous disposez d'un nombre suffisant de tâches pour gérer la charge de travail de votre application. Pendant que votre application évolue en fonction de la demande, la capacité actuelle doit respecter le nombre minimum et maximum de tâches défini par votre action planifiée.

Vous pouvez configurer le dimensionnement du calendrier à l'aide du AWS CLI. Pour plus d'informations sur le dimensionnement planifié, consultez la section Scheduled Scaling du Guide de l'utilisateur d'Application Auto Scaling.

Création d'une action planifiée pour le dimensionnement automatique du service Amazon ECS

Créez une action planifiée pour qu'Amazon ECS augmente ou diminue le nombre de tâches exécutées par votre service en fonction de la date et de l'heure.

console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page des détails du service apparaît.
4. Choisissez Service auto scaling.

   La page Service Auto Scaling apparaît.
5. Si vous n'avez pas configuré le dimensionnement automatique des services, choisissez Set the number of tasks.

   La section sur le nombre de tâches du service Amazon ECS apparaît.

   Sous Nombre de tâches de service Amazon ECS, choisissez Utiliser le dimensionnement automatique du service pour ajuster le nombre de tâches souhaité pour votre service.

   La section Nombre de tâches apparaît.

   a. Dans le champ Nombre minimum de tâches, entrez la limite inférieure du nombre de tâches à utiliser pour le dimensionnement automatique du service. Le nombre souhaité ne sera pas inférieur à ce nombre.
   b. Pour Maximum, entrez la limite supérieure du nombre de tâches à utiliser par Service Auto Scaling. Le nombre souhaité ne sera pas supérieur à ce nombre.
   c. Choisissez Choisir Enregistrer.

      La page des politiques apparaît.
6. Choisissez Actions planifiées, puis sélectionnez Créer.

La page Créer une action planifiée apparaît.

7. Pour Nom de l'action, entrez un nom unique.
8. Dans le champ Fuseau horaire, choisissez un fuseau horaire.

   Tous les fuseaux horaires répertoriés proviennent de la base de données des fuseaux horaires IANA. Pour plus d'informations, voir Liste des fuseaux horaires de la base de données tz.
9. Pour Heure de début, entrez la date et l'heure de début de l'action.

   Si vous avez choisi une planification récurrente, l'heure de début définit le moment où la première action planifiée de la série récurrente s'exécute.
10. Pour Recurrence (Récurrence), choisissez l'une des options disponibles.

    - Pour effectuer une mise à l'échelle selon un calendrier récurrent, choisissez la fréquence à laquelle Amazon ECS exécute l'action planifiée.
      - Si vous choisissez une option qui commence par Rate, l'expression cron est créée pour vous.
      - Si vous sélectionnez Cron, tapez une expression Cron qui spécifie l'heure à laquelle exécuter l'action.
    - Pour effectuer une seule mise à l'échelle, choisissez Une seule fois.
11. Sous Réglages des tâches, procédez comme suit :

    - Dans le champ Minimum, entrez le nombre minimum de tâches que le service doit exécuter.
    - Pour Maximum, entrez le nombre maximum de tâches que le service doit exécuter.
12. Choisissez Create Scheduled Action (Créer une action planifiée).

INTERFACE DE LIGNE DE COMMANDE (CLI)

Utilisez ce qui AWS CLI suit pour configurer les politiques de dimensionnement planifiées pour votre service. Remplacez chaque *`user input placeholder`* par vos propres informations.

Exemple : Pour planifier un dimensionnement unique

Utilisez la put-scheduled-actioncommande suivante avec les `--MaxCapacity` options `--start-time "YYYY-MM-DDThh:mm:ssZ"` et `--MinCapacity` et ou les deux.

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
  --resource-id service/my-cluster/my-service \
```

```
  --scheduled-action-name my-one-time-schedule \
  --start-time 2021-01-30T12:00:00 \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

Exemple : pour planifier le dimensionnement selon un calendrier récurrent

Utilisez la commande put-scheduled-action suivante. Remplacez le *user input* par vos valeurs.

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
  --resource-id service/my-cluster/my-service \
  --scheduled-action-name my-recurring-action \
  --schedule "rate(5 hours)" \
  --start-time 2021-01-30T12:00:00 \
  --end-time 2021-01-31T22:00:00 \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

Le calendrier de récurrence spécifié est basé sur le fuseau horaire UTC. Pour spécifier un autre fuseau horaire, incluez l'`--time-zone`option et le nom du fuseau horaire IANA, comme dans l'exemple suivant.

```
--time-zone "America/New_York"
```

Pour plus d'informations, voir Liste des fuseaux horaires de la base de données tz.

Utilisez des modèles historiques pour faire évoluer les services Amazon ECS grâce à une mise à l'échelle prédictive

La mise à l'échelle prédictive examine les données de charge passées issues des flux de trafic afin d'analyser les tendances quotidiennes ou hebdomadaires. Il utilise ensuite cette analyse pour anticiper les besoins futurs et augmenter de manière proactive les tâches de votre service en fonction des besoins.

La mise à l'échelle automatique prédictive est particulièrement utile dans les situations suivantes.

- Trafic cyclique - Utilisation accrue des ressources pendant les heures normales de bureau et diminution de l'utilisation des ressources le soir et le week-end.
- Modèles on-and-off de charge de travail récurrents - Les exemples incluent le traitement par lots, les tests ou l'analyse périodique des données.
- Applications dont les temps d'initialisation sont longs - Cela peut avoir un impact sur les performances des applications lors d'événements de scale-out, entraînant une latence notable.

Si l'initialisation de vos applications prend du temps et que le trafic augmente régulièrement, vous devriez envisager d'utiliser un dimensionnement prédictif. Il vous aide à évoluer plus rapidement en augmentant de manière proactive le nombre de tâches pour les charges prévues, au lieu d'utiliser des politiques de dimensionnement dynamiques, telles que le suivi des cibles ou le dimensionnement par étapes uniquement. En vous aidant à éviter la possibilité de surprovisionner le nombre de tâches, la mise à l'échelle prédictive peut également vous faire économiser de l'argent.

Prenons l'exemple d'une application très utilisée pendant les heures de bureau et peu utilisée la nuit. Au début de chaque jour ouvrable, la mise à l'échelle prédictive permet d'étendre les tâches avant le premier afflux de trafic. Cela permet à votre application de maintenir une disponibilité et des performances élevées lorsqu'elle passe d'une période de faible utilisation à une période d'utilisation plus intensive. Vous n'avez pas besoin d'attendre que la mise à l'échelle dynamique réagisse à l'évolution du trafic. Vous n'avez pas non plus à perdre du temps à examiner les modèles de charge de votre application et à essayer de planifier le bon nombre de tâches à l'aide d'un dimensionnement planifié.

La mise à l'échelle prédictive est une fonctionnalité de niveau service qui permet d'adapter la tâche de votre service indépendamment de la mise à l'échelle de la capacité de calcul sous-jacente (par exemple, ou EC2 Fargate). Pour Fargate AWS , gère et adapte automatiquement la capacité sous-jacente en fonction des exigences des tâches. Pour ce qui est de la EC2 capacité, vous pouvez utiliser les fournisseurs de capacité du groupe Auto Scaling pour dimensionner automatiquement les EC2 instances sous-jacentes en fonction des exigences de dimensionnement de vos tâches.

Table des matières

- Comment fonctionne le dimensionnement prédictif dans Amazon ECS
- Créez une politique de dimensionnement prédictif pour le dimensionnement automatique du service Amazon ECS
- Évaluez vos politiques de dimensionnement prédictif pour Amazon ECS
- Utiliser des actions planifiées pour remplacer les valeurs de prévision pour Amazon ECS
- Politique de dimensionnement prédictive avancée utilisant des métriques personnalisées pour Amazon ECS

Comment fonctionne le dimensionnement prédictif dans Amazon ECS

Vous trouverez ici des informations sur les considérations relatives à l'utilisation de la mise à l'échelle prédictive, son fonctionnement et ses limites.

Considérations relatives à l'utilisation du dimensionnement prédictif

- Vous devez vous assurer que la mise à l'échelle prédictive est adaptée à votre charge de travail. Vous pouvez vérifier cela en configurant les politiques de dimensionnement en mode prévisions uniquement et en consultant les recommandations de la console. Vous devez évaluer les prévisions et les recommandations avant de commencer à utiliser le dimensionnement prédictif.
- Avant que la mise à l'échelle prédictive puisse commencer à prévoir, elle a besoin d'au moins 24 heures de données historiques. Plus les données historiques disponibles sont nombreuses, plus les prévisions sont efficaces, l'idéal étant de deux semaines. Vous devrez également attendre 24 heures avant que le dimensionnement prédictif puisse générer de nouvelles prévisions lorsque vous supprimez un service Amazon ECS pour en créer un nouveau. L'un des moyens d'accélérer ce processus consiste à utiliser des métriques personnalisées pour agréger les métriques entre l'ancien et le nouveau service Amazon ECS.
- Choisissez une métrique de charge qui représente avec précision la charge totale de votre application et qui constitue l'aspect le plus important de votre application à prendre en compte.
- La mise à l'échelle dynamique associée à la mise à l'échelle prédictive vous aide à suivre de près la demande pour votre application, afin que vous puissiez évoluer pendant les périodes d'accalmie et augmenter votre capacité en cas d'augmentation imprévue du trafic. Lorsque plusieurs politiques de dimensionnement sont actives, chaque stratégie détermine le nombre de tâches souhaité indépendamment, et le nombre de tâches souhaité est défini au maximum de celles-ci.
- Vous pouvez utiliser le dimensionnement prédictif parallèlement à vos politiques de dimensionnement dynamiques, telles que le suivi des cibles ou le dimensionnement par étapes, afin que vos applications évoluent en fonction de modèles historiques et en temps réel. En soi, la mise à l'échelle prédictive ne permet pas d'étendre vos tâches.
- Si vous utilisez un rôle personnalisé lorsque vous appelez l'`register-scalable-target`API, un message d'erreur peut s'afficher indiquant que la politique de dimensionnement prédictif ne peut fonctionner que si le SLR est activé. Dans ce cas, vous devez appeler `register-scalable-target` à nouveau, mais sans role-arn. Utilisez le SLR lors de l'enregistrement de la cible évolutive et appelez l'`put-scaling-policy`API.

Fonctionnement de la mise à l'échelle prédictive

Vous utilisez la mise à l'échelle prédictive en créant une politique de mise à l'échelle prédictive qui spécifie la CloudWatch métrique à surveiller et à analyser. La mise à l'échelle prédictive doit disposer d'au moins 24 heures de données pour commencer à prévoir les valeurs futures.

Une fois que vous avez créé la politique, le dimensionnement prédictif commence à analyser les données métriques recueillies au cours des 14 derniers jours afin d'identifier des modèles. Cette analyse est utilisée pour générer des prévisions horaires des besoins pour les 48 prochaines heures. Les CloudWatch données les plus récentes sont utilisées pour mettre à jour les prévisions toutes les six heures. À mesure que de nouvelles données arrivent, la mise à l'échelle prédictive améliore continuellement la précision des prévisions futures.

Lorsque vous activez la mise à l'échelle prédictive pour la première fois, elle s'exécute en mode prévision uniquement. Il génère des prévisions dans ce mode, mais il n'adapte pas votre service Amazon ECS en fonction de ces prévisions. Cela signifie que vous pouvez évaluer la précision et la pertinence des prévisions. Vous pouvez consulter les données de prévision à l'aide de l'opération `GetPredictiveScalingForecast` API ou du AWS Management Console.

Lorsque vous décidez de commencer à utiliser le dimensionnement prédictif, passez la politique de dimensionnement en mode prévision et échelle. Ce qui suit se produit dans ce mode.

Votre service Amazon ECS est redimensionné au début de chaque heure en fonction des prévisions pour cette heure, par défaut. Vous pouvez choisir de commencer plus tôt en utilisant la `SchedulingBufferTime` propriété dans l'opération `PutScalingPolicy` d'API. Cela permet de lancer de nouvelles tâches avant la demande prévue et leur donne le temps de démarrer et de se préparer à gérer le trafic.

Limite maximale de tâches

Lorsque vous enregistrez les services Amazon ECS pour le dimensionnement, vous définissez un nombre maximum de tâches pouvant être lancées par service. Par défaut, lorsque des politiques de dimensionnement sont définies, elles ne peuvent pas augmenter le nombre de tâches au-delà de sa limite maximale.

Vous pouvez également autoriser l'augmentation automatique du nombre maximum de tâches du service si les prévisions approchent ou dépassent le nombre maximum de tâches du service Amazon ECS.

> **Warning**
>
> Soyez prudent lorsque vous autorisez l'augmentation automatique du nombre maximum de tâches. Cela peut entraîner le lancement d'un plus grand nombre de tâches que prévu, si le nombre maximum de tâches augmenté n'est pas surveillé et géré. Le nombre maximum de tâches augmenté devient alors le nouveau nombre maximum normal de tâches pour

le service Amazon ECS jusqu'à ce que vous le mettiez à jour manuellement. Le nombre maximum de tâches ne revient pas automatiquement au maximum initial.

Régions prises en charge

- USA Est (Virginie du Nord)
- USA Est (Ohio)
- USA Ouest (Californie du Nord)
- USA Ouest (Oregon)
- Afrique (Le Cap)
- Asie-Pacifique (Hong Kong)
- Asie-Pacifique (Jakarta)
- Asie-Pacifique (Mumbai)
- Asie-Pacifique (Osaka)
- Asia Pacific (Seoul)
- Asie-Pacifique (Singapour)
- Asie-Pacifique (Sydney)
- Asie-Pacifique (Tokyo)
- Canada (Centre)
- Chine (Beijing)
- China (Ningxia)
- Europe (Francfort)
- Europe (Irlande)
- Europe (Londres)
- Europe (Milan)
- Europe (Paris)
- Europe (Stockholm)
- Moyen-Orient (Bahreïn)
- Amérique du Sud (São Paulo)
- AWS GovCloud (USA Est)

- AWS GovCloud (US-Ouest)

Créez une politique de dimensionnement prédictif pour le dimensionnement automatique du service Amazon ECS

Créez une politique de dimensionnement prédictive pour qu'Amazon ECS augmente ou diminue le nombre de tâches exécutées par votre service en fonction des données historiques.

> **Note**
>
> Un nouveau service doit fournir au moins 24 heures de données avant de pouvoir générer une prévision.

console

1. Outre les autorisations IAM standard pour créer et mettre à jour des services, vous avez besoin d'autorisations supplémentaires. Pour de plus amples informations, veuillez consulter Autorisations IAM requises pour le dimensionnement automatique du service Amazon ECS.
2. Déterminez les mesures à utiliser pour la politique. Les mesures suivantes sont disponibles :

   - ECSServiceMoyenne CPUUtilization : utilisation moyenne du processeur que le service doit utiliser.
   - ECSServiceAverageMemoryUtilization— Utilisation moyenne de la mémoire que le service doit utiliser.
   - ALBRequestCountPerTarget— Le nombre moyen de demandes par minute que cette tâche devrait idéalement recevoir.

   Vous pouvez également utiliser une métrique personnalisée. Vous devez définir les valeurs suivantes :

   - Charge : métrique qui représente avec précision la charge complète de votre application et qui constitue l'aspect le plus important de votre application à prendre en compte.

   - Mesure de mise à l'échelle : le meilleur indicateur du taux d'utilisation idéal pour votre application.
3. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.

4. Sur la page Clusters, choisissez le cluster.
5. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page des détails du service apparaît.
6. Choisissez Service auto scaling, puis sélectionnez Définir le nombre de tâches.
7. Sous Nombre de tâches de service Amazon ECS, sélectionnez Utiliser le dimensionnement automatique.

   La section Nombre de tâches apparaît.

   a. Dans le champ Nombre minimum de tâches, entrez la limite inférieure du nombre de tâches à utiliser pour le dimensionnement automatique du service. Le nombre souhaité ne sera pas inférieur à ce nombre.
   b. Pour Maximum, entrez la limite supérieure du nombre de tâches à utiliser par Service Auto Scaling. Le nombre souhaité ne sera pas supérieur à ce nombre.
   c. Choisissez Enregistrer.

      La page des politiques apparaît.
8. Choisissez Créer une politique de dimensionnement.

   La page Créer une politique apparaît.
9. Pour le type de politique de mise à l'échelle, choisissez Predictive Scaling.
10. Pour Policy name (Nom de la politique), saisissez un nom de politique.
11. Pour la paire de métriques, choisissez vos métriques dans la liste des options.

    Si vous avez choisi Nombre de demandes Application Load Balancer par cible, choisissez un groupe cible dans le champ Groupe cible. Le nombre de demandes d'Application Load Balancer par cible n'est pris en charge que si vous avez associé un groupe cible Application Load Balancer à votre service.

    Si vous avez choisi une paire de métriques personnalisée, choisissez des métriques individuelles dans les listes pour la métrique de charge et la métrique de mise à l'échelle.
12. Pour Utilisation cible, entrez la valeur cible pour le pourcentage de tâches qu'Amazon ECS doit gérer. Le dimensionnement automatique du service augmente votre capacité jusqu'à ce que l'utilisation moyenne atteigne le taux d'utilisation cible ou jusqu'à ce qu'elle atteigne le nombre maximum de tâches que vous avez spécifié.
13. Choisissez Créer une politique de dimensionnement.

AWS CLI

Utilisez ce qui AWS CLI suit pour configurer des politiques de dimensionnement prédictif pour votre service Amazon ECS. Remplacez chaque *user input placeholder* par vos propres informations.

Pour plus d'informations sur les CloudWatch métriques que vous pouvez spécifier, consultez PredictiveScalingMetricSpecificationle Amazon EC2 Auto Scaling API Reference.

Exemple 1 : politique de dimensionnement prédictif avec mémoire prédéfinie.

Voici un exemple de politique avec une configuration de mémoire prédéfinie.

```
cat policy.json
{
    "MetricSpecifications": [
        {
            "TargetValue": 40,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ECSServiceMemoryUtilization"
            }
        }
    ],
    "SchedulingBufferTime": 3600,
    "MaxCapacityBreachBehavior": "HonorMaxCapacity",
    "Mode": "ForecastOnly"
}
```

L'exemple suivant illustre la création de la politique en exécutant la put-scaling-policycommande avec le fichier de configuration spécifié.

```
aws application-autoscaling put-scaling-policy \
--service-namespace ecs \
--region us-east-1 \
--policy-name predictive-scaling-policy-example \
--resource-id service/MyCluster/test \
--policy-type PredictiveScaling \
--scalable-dimension ecs:service:DesiredCount \
--predictive-scaling-policy-configuration file://policy.json
```

En cas de succès, cette commande renvoie l'ARN de la politique.

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
east-1:012345678912:scalingPolicy:d1d72dfe-5fd3-464f-83cf-824f16cb88b7:resource/ecs/
service/MyCluster/test:policyName/predictive-scaling-policy-example",
    "Alarms": []
}
```

Exemple 2 : politique de dimensionnement prédictif avec processeur prédéfini.

Voici un exemple de politique avec une configuration de processeur prédéfinie.

```
cat policy.json
{
    "MetricSpecifications": [
        {
            "TargetValue": 0.00000004,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ECSServiceCPUUtilization"
            }
        }
    ],
    "SchedulingBufferTime": 3600,
    "MaxCapacityBreachBehavior": "HonorMaxCapacity",
    "Mode": "ForecastOnly"
}
```

L'exemple suivant illustre la création de la politique en exécutant la put-scaling-policycommande avec le fichier de configuration spécifié.

```
aws aas put-scaling-policy \
--service-namespace ecs \
--region us-east-1 \
--policy-name predictive-scaling-policy-example \
--resource-id service/MyCluster/test \
--policy-type PredictiveScaling \
--scalable-dimension ecs:service:DesiredCount \
--predictive-scaling-policy-configuration file://policy.json
```

En cas de succès, cette commande renvoie l'ARN de la politique.

```
{
```

```
    "PolicyARN": "arn:aws:autoscaling:us-
east-1:012345678912:scalingPolicy:d1d72dfe-5fd3-464f-83cf-824f16cb88b7:resource/ecs/
service/MyCluster/test:policyName/predictive-scaling-policy-example",
    "Alarms": []
}
```

Évaluez vos politiques de dimensionnement prédictif pour Amazon ECS

Avant d'utiliser une politique de dimensionnement prédictif pour dimensionner vos services, consultez les recommandations et les autres données relatives à votre politique dans la console Amazon ECS. C'est une étape importante pour éviter qu'une politique de mise à l'échelle prédictive mette votre capacité réelle à l'échelle tant que vous ne savez pas si ses prévisions sont exactes.

S'il s'agit d'un nouveau service, comptez 24 heures pour créer la première prévision.

Lors de AWS la création d'une prévision, elle utilise des données historiques. Si votre service ne dispose pas encore de beaucoup de données historiques récentes, le dimensionnement prédictif peut temporairement remplacer les prévisions par des agrégats créés à partir des agrégats historiques actuellement disponibles. Les prévisions sont remplies jusqu'à deux semaines avant la date de création d'une politique.

Afficher vos recommandations de mise à l'échelle prédictive

Pour une analyse efficace, le dimensionnement automatique des services doit comporter au moins deux politiques de dimensionnement prédictif à comparer. (Toutefois, vous pouvez toujours consulter les résultats d'une seule politique.) Lorsque vous créez plusieurs politiques, vous pouvez évaluer une politique qui utilise une seule métrique par rapport à une autre qui utilise une autre métrique. Vous pouvez également évaluer l'impact de différentes combinaisons de valeurs cibles et de métriques. Une fois les politiques de dimensionnement prédictif créées, Amazon ECS commence immédiatement à évaluer quelle politique serait la plus efficace pour dimensionner votre groupe.

Pour consulter vos recommandations dans la console Amazon ECS

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page des détails du service apparaît.
4. Choisissez Service auto scaling.

5. Choisissez la politique de dimensionnement prédictif, puis sélectionnez Actions, Predictive Scaling, Afficher les recommandations.

   Vous pouvez consulter les détails d'une politique ainsi que nos recommandations. La recommandation vous indique si la politique de mise à l'échelle prédictive est plus efficace que si vous ne l'utilisez pas.

   Si vous ne savez pas si une politique de mise à l'échelle prédictive convient à votre groupe, consultez les colonnes Impact sur la disponibilité et Impact sur les coûts pour choisir la politique appropriée. Les informations de chaque colonne vous indiquent l'impact de la politique.

   - Impact sur la disponibilité : indique si la politique permettrait d'éviter un impact négatif sur la disponibilité en provisionnant suffisamment de tâches pour gérer la charge de travail, par rapport à la non-utilisation de la stratégie.
   - Impact sur les coûts : indique si la politique éviterait un impact négatif sur vos coûts en évitant de surprovisionner les tâches, par rapport à la non-utilisation de la politique. En surprovisionnant, vos services sont sous-utilisés ou inactifs, ce qui ne fait qu'aggraver l'impact sur les coûts.

   Si vous avez plusieurs politiques, la balise Meilleure prévision s'affiche à côté du nom de la politique qui offre le plus d'avantages en matière de disponibilité à moindre coût. Une plus grande importance est accordée à l'impact sur la disponibilité.

6. (Facultatif) Pour sélectionner la période souhaitée pour les résultats des recommandations, choisissez votre valeur préférée dans le menu déroulant Période d'évaluation : 2 jours, 1 semaine ou 2 semaines. Par défaut, la période d'évaluation est réglée sur les deux dernières semaines. Une période d'évaluation plus longue fournit davantage de points de données pour les résultats des recommandations. Toutefois, l'ajout de points de données supplémentaires risque de ne pas améliorer les résultats si vos modèles de charge ont changé, par exemple après une période de demande exceptionnelle. Dans ce cas, vous pouvez obtenir une recommandation plus ciblée en consultant des données plus récentes.

> **Note**
>
> Les recommandations sont générées uniquement pour les politiques qui sont en mode Prévision uniquement. La fonctionnalité des recommandations offre de meilleurs résultats lorsqu'une politique est en mode Prévision uniquement pendant toute la période d'évaluation.

Si vous lancez une politique en mode Prévision et mise à l'échelle et que vous la passez ultérieurement en mode Prévision uniquement, les résultats de cette politique risquent d'être biaisés. Cela s'explique par le fait que la politique a déjà contribué à la capacité réelle.

Consulter les graphiques de surveillance de la mise à l'échelle prédictive

Dans la console, vous pouvez consulter les prévisions des jours, semaines ou mois précédents pour visualiser les performances de la politique au fil du temps. Vous pouvez également utiliser ces informations pour évaluer l'exactitude des prévisions lorsque vous décidez de laisser une politique redimensionner votre nombre réel de tâches.

Pour consulter les graphiques de surveillance du dimensionnement prédictif dans la console Amazon ECS

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page des détails du service apparaît.
4. Choisissez Service auto scaling.
5. Choisissez la politique de dimensionnement prédictif, puis sélectionnez Actions, Predictive Scaling, View Graph.
6. Dans la section Surveillance, vous pouvez afficher les prévisions passées et futures de votre politique concernant la charge et la capacité par rapport aux valeurs réelles. Le graphique Charge présente la prévision de charge et les valeurs réelles pour la métrique de charge que vous avez choisie. Le graphique de capacité indique le nombre de tâches prévues par la politique. Il inclut également le nombre réel de tâches lancées. La ligne verticale sépare les valeurs historiques des prévisions futures. Ces graphiques sont disponibles peu de temps après la création de la politique.
7. (Facultatif) Pour modifier la quantité de données historiques affichées dans le graphique, choisissez la valeur de votre choix dans la liste déroulante Période d'évaluation en haut de la page. La période d'évaluation ne transforme en rien les données de cette page. Elle ne fait que modifier la quantité de données historiques affichées.

Comparer les données du graphique Charge

Chaque ligne horizontale représente un ensemble différent de points de données rapportés à des intervalles d'une heure :

1. La charge observée réelle utilise la statistique SUM correspondant à la métrique de charge de votre choix afin d'afficher la charge horaire totale dans le passé.
2. La charge prévue par la politique indique la prévision de charge horaire. Cette prévision se base sur les observations de charge réelles des deux semaines précédentes.

Comparez les données du graphique Capacité

Chaque ligne horizontale représente un ensemble différent de points de données rapportés à des intervalles d'une heure :

1. Le nombre réel de tâches observées indique la capacité réelle de votre service Amazon ECS dans le passé, qui dépend de vos autres politiques de dimensionnement et de la taille de groupe minimale en vigueur pour la période sélectionnée.
2. La capacité prévue par la politique indique la capacité de base à laquelle vous pouvez vous attendre au début de chaque heure lorsque la politique est en mode Prévision et mise à l'échelle.
3. Le nombre de tâches requis déduit indique le nombre idéal de tâches dans votre service pour maintenir l'indicateur de mise à l'échelle à la valeur cible que vous avez choisie.
4. Le nombre minimum de tâches indique le nombre minimum de tâches de votre service.
5. La capacité maximale indique le nombre maximal de tâches de votre service.

Pour calculer la capacité requise déduite, nous partons du principe que chaque tâche est utilisée de manière égale à une valeur cible spécifiée. Dans la pratique, le nombre de tâches n'est pas utilisé de la même manière. En supposant que l'utilisation est uniformément répartie entre les tâches, nous pouvons toutefois faire une estimation probable de la quantité de capacité nécessaire. Le nombre de tâches requis est ensuite calculé de manière à être inversement proportionnel à la métrique de mise à l'échelle que vous avez utilisée pour votre politique de dimensionnement prédictive. En d'autres termes, à mesure que le nombre de tâches augmente, la métrique de mise à l'échelle diminue au même rythme. Par exemple, si le nombre de tâches double, la métrique de mise à l'échelle doit être réduite de moitié.

La formule de la capacité requise déduite est la suivante :

```
sum of (actualServiceUnits*scalingMetricValue)/(targetUtilization)
```

Par exemple, nous prenons les `actualServiceUnits` (`10`) et la `scalingMetricValue` (`30`) pour une heure donnée. Nous prenons ensuite la `targetUtilization` que vous avez spécifiée dans votre politique de mise à l'échelle prédictive (`60`) et calculons la capacité requise déduite pour la même heure. Elle renvoie la valeur `5`. Cela signifie que cinq est la quantité de capacité requise déduite pour maintenir la capacité en proportion inverse directe à la valeur cible de la métrique de mise à l'échelle.

> **Note**
>
> Différents leviers sont à votre disposition pour ajuster et améliorer les économies de coûts et la disponibilité de votre application.
>
> - Vous utilisez la mise à l'échelle prédictive pour la capacité de base et la mise à l'échelle dynamique afin de gérer la capacité supplémentaire. La mise à l'échelle dynamique fonctionne indépendamment de la mise à l'échelle prédictive, avec une mise à l'échelle horizontale et une montée en puissance en fonction de l'utilisation actuelle. Amazon ECS calcule d'abord le nombre de tâches recommandé pour chaque politique de dimensionnement non planifiée. Il évolue ensuite en fonction de la politique qui fournit le plus grand nombre de tâches.
> - Pour permettre une mise à l'échelle lorsque la charge diminue, votre service doit toujours disposer d'au moins une politique de dimensionnement dynamique avec la partie évolutive activée.
> - Vous pouvez améliorer les performances de mise à l'échelle en vous assurant que vos capacités minimale et maximale ne sont pas trop restrictives. Une politique avec un nombre recommandé de tâches qui ne se situe pas dans la plage de capacité minimale et maximale ne pourra pas être étendue à l'entrée et à la sortie.

Surveillez les mesures de dimensionnement prédictif pour Amazon ECS avec CloudWatch

Vous pouvez utiliser Amazon CloudWatch pour surveiller vos données à des fins de dimensionnement prédictif. Une politique de dimensionnement prédictif collecte des données qui sont utilisées pour prévoir votre future charge. Les données collectées sont automatiquement stockées CloudWatch à intervalles réguliers et peuvent être utilisées pour visualiser les performances de la politique au fil du temps. Vous pouvez également créer des CloudWatch alarmes pour vous avertir lorsque les indicateurs de performance changent au-delà des limites que vous avez définies.

Visualisez les données de prévision historiques

Les données de prévision de charge pour une politique de dimensionnement prédictif peuvent être consultées CloudWatch et peuvent être utiles lors de la visualisation des prévisions par rapport à d'autres CloudWatch indicateurs dans un seul graphique. Vous pouvez également voir les tendances au fil du temps en consultant une plage de temps plus large. Vous pouvez accéder aux métriques historiques jusqu'à 15 mois pour acquérir un meilleur point de vue de la façon dont votre politique s'exécute.

Pour consulter les données de prévision historiques à l'aide de la CloudWatch console

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, choisissez Metrics (Métriques), All metrics (Toutes les métriques).
3. Choisissez l'espace de noms de métriques Application Auto Scaling.
4. Choisissez Predictive Scaling Load Forecasts.
5. Dans le champ de recherche, entrez le nom de la politique de dimensionnement prédictif ou le nom du groupe de services Amazon ECS, puis appuyez sur Entrée pour filtrer les résultats.
6. Pour représenter graphiquement une métrique, cochez la case en regard de la métrique. Pour modifier le nom du graphique, choisissez l'icône représentant un crayon. Pour modifier la plage de temps, sélectionnez l'une des valeurs prédéfinies ou choisissez custom (personnalisé). Pour plus d'informations, consultez la section Représentation graphique d'une métrique dans le guide de CloudWatch l'utilisateur Amazon.
7. Pour modifier les statistiques, choisissez l'onglet Graphed metrics (Graphique des métriques). Sélectionnez l'en-tête de colonne ou une valeur individuelle et choisissez une autre valeur. Bien que vous puissiez choisir n'importe quelle statistique pour chaque métrique, toutes les statistiques ne sont pas utiles pour les PredictiveScalingLoadForecastmétriques. Par exemple, les calculs statistiques de moyenne, minimum et maximum de l'utilisation de l'UC sont utiles, mais le calcul statistique de somme ne l'est pas.
8. (En option) Pour ajouter une autre métrique à utiliser dans l'expression mathématique, sous Toutes les métriques, choisissez Tous, recherchez la métrique spécifique, puis activez la case à cocher en regard de celle-ci. Vous pouvez ajouter jusqu'à 10 métriques.
9. (Facultatif) Pour ajouter le graphique à un CloudWatch tableau de bord, choisissez Actions, puis Ajouter au tableau de bord.

Créer des mesures de précision à l'aide de mathématiques

Avec les mathématiques métriques, vous pouvez interroger plusieurs CloudWatch métriques et utiliser des expressions mathématiques pour créer de nouvelles séries chronologiques basées sur ces métriques. Vous pouvez visualiser les séries chronologiques obtenues sur la CloudWatch console et les ajouter aux tableaux de bord. Pour plus d'informations sur les mathématiques métriques, consultez la section Utilisation des mathématiques métriques dans le guide de CloudWatch l'utilisateur Amazon.

À l'aide des mathématiques métriques, vous pouvez représenter graphiquement les données générées par le dimensionnement automatique des services à des fins de dimensionnement prédictif de différentes manières. Cela vous permet de surveiller les performances des politiques au fil du temps et de comprendre si votre combinaison de mesures peut être améliorée.

Par exemple, vous pouvez utiliser une expression mathématique de métrique pour surveillererreur absolue moyenne en pourcentage(MÂLE). La métrique MAPE permet de surveiller la différence entre les valeurs prévues et les valeurs réelles observées pendant une fenêtre de prévision donnée. Les modifications de la valeur de MAPE peuvent indiquer si les performances de la stratégie se dégradent au fil du temps à mesure que la nature de votre application change. Une augmentation de l'EMAP indique un écart plus important entre les valeurs prévues et les valeurs réelles.

Exemple : expressions mathématiques appliquées aux métriques

Pour commencer avec ce type de graphique, vous pouvez créer une expression mathématique de métrique comme celle présentée dans l'exemple suivant.

Au lieu d'une seule métrique, il existe un tableau de structures de requêtes de données de métriques pour`MetricDataQueries`. Chaque élément de`MetricDataQueries`obtient une métrique ou exécute une expression mathématique. Le premier point,`e1`, est l'expression mathématique. L'expression désignée définit le`ReturnData`Paramètre to`true`, qui génère une seule série chronologique. Pour toutes les autres métriques, le kit`ReturnData`valeur est`false`.

Dans l'exemple, l'expression désignée utilise les valeurs réelles et prévues comme entrée et renvoie la nouvelle métrique (MAPE). `m1`est la CloudWatch métrique qui contient les valeurs de charge réelles (en supposant que l'utilisation du processeur est la métrique de charge initialement spécifiée pour la politique nommée`my-predictive-scaling-policy`). `m2`est la CloudWatch métrique qui contient les valeurs de charge prévues. La syntaxe mathématique de la métrique MAPE est la suivante :

Moyenne de (abs ((Réel - Forecast)/(Réel)))

Visualisez vos mesures de précision et définissez des alarmes

Pour visualiser les données métriques de précision, sélectionnez l'onglet Métriques dans la CloudWatch console. Vous pouvez représenter les données sous forme graphique à partir de là. Pour plus d'informations, consultez la section Ajouter une expression mathématique à un CloudWatch graphique dans le guide de CloudWatch l'utilisateur Amazon.

Vous pouvez définir une alarme sur une métrique que vous surveillez à partir de la section Metrics (Métriques). Pendant que vous êtes sur l'onglet Métriques sous forme de graphique, vous pouvez sélectionner l'icône Créer une alarme sous la colonne Actions. LeCréer une alarmeest représentée par une petite cloche. Pour plus d'informations et pour connaître les options de notification, consultez les sections Création CloudWatch d'une alarme basée sur une expression mathématique métrique et Notification aux utilisateurs des modifications apportées aux alarmes dans le guide de l' CloudWatch utilisateur Amazon.

Vous pouvez également utiliser GetMetricDataet effectuer des calculs PutMetricAlarmà l'aide des mathématiques métriques et créer des alarmes en fonction de la sortie.

Utiliser des actions planifiées pour remplacer les valeurs de prévision pour Amazon ECS

Parfois, vous pouvez disposer d'informations supplémentaires sur de futurs besoins de votre application que le calcul prédictif ne peut pas prendre en compte. Par exemple, les calculs de prévisions peuvent sous-estimer les tâches nécessaires pour un événement marketing à venir. Vous pouvez alors utiliser des actions planifiées pour remplacer temporairement la prévision au cours de périodes ultérieures. Les actions planifiées peuvent être exécutées de manière récurrente, ou à une date et une heure spécifiques en cas de fluctuations ponctuelles de la demande.

Par exemple, vous pouvez créer une action planifiée comportant un nombre de tâches supérieur au nombre prévu. Au moment de l'exécution, Amazon ECS met à jour le nombre minimum de tâches de votre service. La mise à l'échelle prédictive étant optimisée en fonction du nombre de tâches, une action planifiée comportant un nombre minimum de tâches supérieur aux valeurs prévisionnelles est respectée. Cela permet d'éviter que le nombre de tâches ne soit inférieur aux prévisions. Pour ne plus annuler les prévisions, utilisez une deuxième action planifiée pour rétablir le nombre minimum de tâches dans ses paramètres d'origine.

La procédure suivante présente les étapes à suivre pour remplacer la prévision au cours de périodes ultérieures.

Rubriques

- Étape 1 : (facultatif) Analyser les données en séries chronologiques

- Étape 2 : créer deux actions planifiées

> **Important**
>
> Cette rubrique part du principe que vous essayez de modifier les prévisions pour vous adapter à une capacité supérieure à celle prévue. Si vous devez réduire temporairement le nombre de tâches sans interférer avec une politique de dimensionnement prédictif, utilisez plutôt le mode prévision uniquement. En mode prévisions uniquement, la mise à l'échelle prédictive continuera de générer des prévisions, mais elle n'augmentera pas automatiquement le nombre de tâches. Vous pouvez ensuite surveiller l'utilisation des ressources et réduire manuellement le nombre de tâches selon les besoins.

Étape 1 : (facultatif) Analyser les données en séries chronologiques

Commencez par analyser les données en séries chronologiques de la prévision. Il s'agit d'une étape facultative, mais elle permet de comprendre les détails de la prévision.

1. Récupérer la prévision

   Une fois la prévision créée, vous pouvez interroger une période spécifique au sein de celle-ci. L'objectif de la requête est d'obtenir une vue complète des données en séries chronologiques d'une période spécifique.

   Votre requête peut inclure jusqu'à deux jours de données de prévision ultérieures. Si vous utilisez la mise à l'échelle prédictive depuis un certain temps, vous pouvez également accéder à vos données de prévision antérieures. Toutefois, la durée maximale entre le début et la fin est de 30 jours.

   Pour obtenir les prévisions à l'aide de la get-predictive-scaling-forecast AWS CLI commande, entrez les paramètres suivants dans la commande :

   - Entrez le nom du cluster dans le `resource-id` paramètre.
   - Entrez le nom de la stratégie dans le paramètre `--policy-name`.
   - Entrez l'heure de début dans le paramètre `--start-time` pour ne renvoyer que les données de prévision correspondant à l'heure spécifiée ou ultérieures à celle-ci.
   - Entrez l'heure de fin dans le paramètre `--end-time` pour ne renvoyer que les données de prévision correspondant à l'heure spécifiée ou antérieures à celle-ci.

```
aws application-autoscaling get-predictive-scaling-forecast \
    --service-namespace ecs \
    --resource-id service/MyCluster/test \
    --policy-name cpu40-predictive-scaling-policy \
    --scalable-dimension ecs:service:DesiredCount \
    --start-time "2021-05-19T17:00:00Z" \
    --end-time "2021-05-19T23:00:00Z"
```

Si elle aboutit, la commande renvoie des données semblables à l'exemple suivant.

```
{
    "LoadForecast": [
        {
            "Timestamps": [
                "2021-05-19T17:00:00+00:00",
                "2021-05-19T18:00:00+00:00",
                "2021-05-19T19:00:00+00:00",
                "2021-05-19T20:00:00+00:00",
                "2021-05-19T21:00:00+00:00",
                "2021-05-19T22:00:00+00:00",
                "2021-05-19T23:00:00+00:00"
            ],
            "Values": [
                153.0655799339254,
                128.8288551285919,
                107.1179447150675,
                197.3601844551528,
                626.4039934516954,
                596.9441277518481,
                677.9675713779869
            ],
            "MetricSpecification": {
                "TargetValue": 40.0,
                "PredefinedMetricPairSpecification": {
                    "PredefinedMetricType": "ASGCPUUtilization"
                }
            }
        }
    ],
    "CapacityForecast": {
        "Timestamps": [
```

```
            "2021-05-19T17:00:00+00:00",
            "2021-05-19T18:00:00+00:00",
            "2021-05-19T19:00:00+00:00",
            "2021-05-19T20:00:00+00:00",
            "2021-05-19T21:00:00+00:00",
            "2021-05-19T22:00:00+00:00",
            "2021-05-19T23:00:00+00:00"
        ],
        "Values": [
            2.0,
            2.0,
            2.0,
            2.0,
            4.0,
            4.0,
            4.0
        ]
    },
    "UpdateTime": "2021-05-19T01:52:50.118000+00:00"
}
```

La réponse comprend deux prévisions : `LoadForecast` et `CapacityForecast`. `LoadForecast` affiche la prévision de charge horaire. `CapacityForecast` affiche les valeurs de prévision de la capacité nécessaire sur une base horaire pour gérer la charge prévue tout en maintenant une `TargetValue` de 40 (40 % d'utilisation moyenne du processeur).

2. Identifier la période cible

   Indiquez l'heure ou les heures où la fluctuation de la demande ponctuelle devrait avoir lieu. N'oubliez pas que les dates et les heures indiquées dans la prévision sont basées sur le fuseau horaire UTC.

Étape 2 : créer deux actions planifiées

Créez ensuite deux actions planifiées pour une période spécifique où votre application devra gérer une charge plus élevée que celle prédite. Par exemple, si vous organisez un événement marketing qui va générer du trafic sur votre site pendant une période limitée, vous pouvez planifier une action ponctuelle pour mettre à jour la capacité minimale au début de cet événement. Puis vous pouvez planifier une autre action pour rétablir le paramètre d'origine de la capacité minimale à la fin de l'événement.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, puis choisissez le service.

   La page des détails du service apparaît.
4. Choisissez Service Auto Scaling.

   La page des politiques apparaît.
5. Choisissez Actions planifiées, puis sélectionnez Créer.

   La page d'action Créer un calendrier apparaît.
6. Pour Nom de l'action, entrez un nom unique.
7. Dans le champ Fuseau horaire, choisissez un fuseau horaire.

   Tous les fuseaux horaires répertoriés proviennent de la base de données des fuseaux horaires IANA. Pour plus d'informations, voir Liste des fuseaux horaires de la base de données tz.
8. Pour Heure de début, entrez la date et l'heure de début de l'action.
9. Pour Recurrence (Récurrence), choisissez Once (Une fois).
10. Sous Réglages des tâches, pour Minimum, entrez une valeur inférieure ou égale au nombre maximum de tâches.
11. Choisissez Create Scheduled Action (Créer une action planifiée).

    La page des politiques apparaît.
12. Configurez une deuxième action planifiée pour rétablir le nombre minimum de tâches au paramètre d'origine à la fin de l'événement. Le dimensionnement prédictif peut ajuster le nombre de tâches uniquement lorsque la valeur que vous définissez pour Minimum est inférieure aux valeurs de prévision.

Pour créer deux actions planifiées pour des événements ponctuels (AWS CLI)

Pour AWS CLI créer les actions planifiées, utilisez la commande put-scheduled-update-group-action.

Par exemple, définissons une action planifiée pour maintenir une capacité minimale de trois instances pendant huit heures à partir du 19 mai à 17h00. Les commandes suivantes montrent comment implémenter ce scénario.

La première commande put-scheduled-update-group-action demande à Amazon EC2 Auto Scaling de mettre à jour la capacité minimale du groupe Auto Scaling spécifié à 17 h 00 UTC le 19 mai 2021.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-start \
  --auto-scaling-group-name my-asg --start-time "2021-05-19T17:00:00Z" --minimum-capacity 3
```

La deuxième commande demande à Amazon EC2 Auto Scaling de définir la capacité minimale du groupe à 1 h 00 UTC le 20 mai 2021.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-end \
  --auto-scaling-group-name my-asg --start-time "2021-05-20T01:00:00Z" --minimum-capacity 1
```

Après avoir ajouté ces actions planifiées au groupe Auto Scaling, Amazon EC2 Auto Scaling effectue les opérations suivantes :

- À 17h00 UTC le 19 mai 2021, la première action planifiée s'exécute. Si le groupe compte actuellement moins de trois instances, il passe à trois instances. Pendant cette période et pendant les huit prochaines heures, Amazon EC2 Auto Scaling peut continuer à évoluer si la capacité prévue est supérieure à la capacité réelle ou si une politique de dimensionnement dynamique est en vigueur.
- À 1h00 UTC le 20 mai 2021, la seconde action planifiée s'exécute. Cette action rétablit le paramètre d'origine de la capacité minimale à la fin de l'événement.

Mise à l'échelle basée sur des planifications récurrentes

Pour remplacer la prévision applicable à la même période chaque semaine, créez deux actions planifiées et fournissez la logique d'heure et de date à l'aide d'une expression cron.

L'expression cron est constituée de cinq champs séparés par des espaces : [Minute] [Heure] [Jour_du_Mois] [Mois_de_Année] [Jour_de_Semaine]. Ces champs peuvent contenir toutes les valeurs autorisées, y compris des caractères spéciaux.

Par exemple, l'expression cron suivante exécute l'action tous les mardis à 6h30. L'astérisque est utilisé comme caractère générique pour représenter toutes les valeurs d'un champ.

```
30 6 * * 2
```

Consultez aussi

Pour plus d'informations sur la gestion des actions planifiées, consultezUtilisez des actions planifiées pour faire évoluer les services Amazon ECS.

Politique de dimensionnement prédictive avancée utilisant des métriques personnalisées pour Amazon ECS

Vous pouvez utiliser des métriques prédéfinies ou personnalisées dans une politique de dimensionnement prédictif. Les métriques personnalisées sont utiles lorsque les métriques prédéfinies (telles que le processeur, la mémoire, etc.) ne sont pas suffisantes pour décrire suffisamment la charge de votre application.

Lorsque vous créez une politique de dimensionnement prédictif avec des métriques personnalisées, vous pouvez spécifier d'autres CloudWatch métriques fournies par AWS. Vous pouvez également spécifier des métriques que vous définissez et publiez vous-même. Vous pouvez également utiliser les mathématiques des métriques pour agréger et transformer les métriques existantes en une nouvelle série chronologique qui AWS n'est pas automatiquement suivie. Par exemple, vous pouvez combiner des valeurs dans vos données en calculant de nouvelles sommes ou moyennes, ce que l'on appelle l'agrégation. Les données résultantes sont appelées un agrégat.

La section suivante contient les bonnes pratiques et des exemples de construction de la structure JSON pour la politique.

Prérequis

Pour ajouter des métriques personnalisées à votre politique de mise à l'échelle, vous devez disposer des autorisations `cloudwatch:GetMetricData`.

Pour spécifier vos propres indicateurs au lieu des indicateurs AWS fournis, vous devez d'abord les publier sur CloudWatch. Pour plus d'informations, consultez la section Publication de métriques personnalisées dans le guide de CloudWatch l'utilisateur Amazon.

Si vous publiez vos propres métriques, veillez à publier les points de données à une fréquence minimale de cinq minutes. Les points de données sont extraits CloudWatch en fonction de la durée de la période dont ils ont besoin. Par exemple, la spécification des métriques de charge utilise des métriques horaires pour mesurer la charge de votre application. CloudWatch utilise vos données métriques publiées pour fournir une valeur de données unique pour toute période d'une heure en

agrégeant tous les points de données avec des horodatages correspondant à chaque période d'une heure.

Bonnes pratiques

Les bonnes pratiques suivantes peuvent vous aider à utiliser plus efficacement les métriques personnalisées :

- La métrique la plus utile pour la spécification de la métrique de charge est une métrique qui représente la charge sur un groupe Auto Scaling dans son ensemble.
- La métrique la plus utile pour la mise à l'échelle d'une spécification de métrique est une métrique de débit moyen ou d'utilisation par tâche.
- L'utilisation cible doit correspondre au type de métrique de mise à l'échelle. Pour une configuration de politique qui utilise l'utilisation du processeur, il s'agit d'un pourcentage cible, par exemple.
- Si ces recommandations ne sont pas suivies, les valeurs futures prédites des séries temporelles seront probablement incorrectes. Pour vérifier que les données sont correctes, vous pouvez consulter les valeurs prévisionnelles dans la console. Sinon, après avoir créé votre politique de dimensionnement prédictif, inspectez les `LoadForecast` objets renvoyés par un appel à l'GetPredictiveScalingForecastAPI.
- Nous vous recommandons vivement de configurer le dimensionnement prédictif en mode prévisions uniquement afin de pouvoir évaluer les prévisions avant que le dimensionnement prédictif ne commence à évoluer activement.

Limites

- Vous pouvez interroger des points de données de 10 métriques au maximum dans une spécification métrique.
- Dans le cadre de cette limite, une expression compte pour une métrique.

Résolution des problèmes liés à une politique de dimensionnement prédictive à l'aide de métriques personnalisées

Si un problème survient lors de l'utilisation de métriques personnalisées, nous vous recommandons d'effectuer les opérations suivantes :

- Si vous rencontrez un problème lors d'un blue/green déploiement lors de l'utilisation d'une expression de recherche, assurez-vous d'avoir créé une expression de recherche qui recherche

une correspondance partielle et non une correspondance exacte. Vous devez également vérifier que la requête ne trouve que les groupes Auto Scaling en cours d'exécution dans l'application spécifique. Pour plus d'informations sur la syntaxe des expressions de recherche, consultez la section Syntaxe des expressions de CloudWatch recherche dans le guide de CloudWatch l'utilisateur Amazon.

- La put-scaling-policycommande valide une expression lorsque vous créez votre politique de dimensionnement. Cependant, il est possible que cette commande ne parvienne pas à identifier la cause exacte des erreurs détectées. Pour résoudre les problèmes, corrigez les erreurs que vous recevez en réponse à une demande de get-metric-datacommande. Vous pouvez également résoudre les problèmes liés à l'expression depuis la CloudWatch console.
- Vous devez spécifier `false` pour `ReturnData` si `MetricDataQueries` spécifie la fonction SEARCH() seule sans une fonction mathématique comme SUM(). Cela est dû au fait que les expressions de recherche peuvent renvoyer plusieurs séries temporelles et qu'une spécification métrique basée sur une expression ne peut renvoyer qu'une seule séries temporelles.
- Toutes les métriques impliquées dans une expression de recherche doivent avoir la même résolution.

Création du JSON pour le dimensionnement prédictif (métriques personnalisées) avec Amazon ECS

La section suivante contient des exemples de configuration de la mise à l'échelle prédictive pour interroger des données CloudWatch. Il existe deux méthodes pour configurer cette option, qui affecteront le format utilisé pour créer le fichier JSON de votre politique de mise à l'échelle prédictive. Lorsque vous utilisez des mathématiques de métriques, le format du fichier JSON varie davantage en fonction des mathématiques de métriques effectuées.

1. Pour créer une politique qui obtient des données directement à partir d'autres CloudWatch indicateurs fournis par AWS ou sur lesquels vous publiez CloudWatch, voirExemple de politique de dimensionnement prédictif avec des mesures de charge et de dimensionnement personnalisées à l'aide du AWS CLI.

Exemple de politique de dimensionnement prédictif avec des mesures de charge et de dimensionnement personnalisées à l'aide du AWS CLI

Pour créer une politique de dimensionnement prédictive avec des métriques de charge et de dimensionnement personnalisées avec le AWS CLI, stockez les arguments pour `--predictive-scaling-configuration` dans un fichier JSON nommé`config.json`.

Vous commencez par ajouter des métriques personnalisées en remplaçant les valeurs remplaçables de l'exemple suivant par celles de vos métriques et de votre utilisation cible.

```
{
  "MetricSpecifications": [
    {
      "TargetValue": 50,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "scaling_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyUtilizationMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Average"
            }
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyLoadMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Sum"
```

```
            }
          }
        ]
      }
    }
  ]
}
```

Pour plus d'informations, consultez MetricDataQueryle manuel Amazon EC2 Auto Scaling API Reference.

> **Note**
>
> Voici quelques ressources supplémentaires qui peuvent vous aider à trouver des noms de métriques, des espaces de noms, des dimensions et des statistiques pour les CloudWatch métriques :
>
> - Pour plus d'informations sur les métriques disponibles pour les AWS services, consultez les AWS services qui publient CloudWatch des métriques dans le guide de CloudWatch l'utilisateur Amazon.
> - Pour obtenir le nom, l'espace de noms et les dimensions exacts (le cas échéant) d'une CloudWatch métrique avec le AWS CLI, consultez list-metrics.

Pour créer cette politique, exécutez la put-scaling-policycommande en utilisant le fichier JSON comme entrée, comme illustré dans l'exemple suivant.

```
aws application-autoscaling put-scaling-policy --policy-name my-predictive-scaling-
policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

Si elle aboutit, cette commande renvoie l'Amazon Resource Name (ARN) de la stratégie.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-predictive-scaling-policy",
  "Alarms": []
}
```

Utiliser des expressions mathématiques de métrique

La section suivante fournit des informations sur l'utilisation des mathématiques métriques avec les politiques de dimensionnement prédictif dans votre politique.

Comprendre les mathématiques de métrique

Si vous souhaitez simplement agréger des données métriques existantes, les mathématiques CloudWatch métriques vous évitent les efforts et les coûts liés à la publication d'une autre métrique dans CloudWatch. Vous pouvez utiliser n'importe quelle métrique qui AWS fournit, et vous pouvez également utiliser des métriques que vous définissez dans le cadre de vos applications.

Pour plus d'informations, consultez la section Utilisation des mathématiques métriques dans le guide de CloudWatch l'utilisateur Amazon.

Si vous choisissez d'utiliser une expression mathématique de métrique dans votre politique de mise à l'échelle prédictive, tenez compte des points suivants :

- Les opérations mathématiques métriques utilisent les points de données d'une combinaison unique de keys/value paires de métriques de nom, d'espace de noms et de dimensions.
- Vous pouvez utiliser n'importe quel opérateur arithmétique (+ - */^), fonction statistique (telle que AVG ou SUM) ou toute autre fonction compatible. CloudWatch
- Vous pouvez utiliser à la fois des métriques et les résultats d'autres expressions mathématiques dans les formules de l'expression mathématique.
- Vos expressions mathématiques de métrique peuvent être composées de différentes agrégations. Cependant, une bonne pratique pour le résultat final de l'agrégation consiste à utiliser `Average` pour la métrique de mise à l'échelle et `Sum` pour la métrique de charge.
- Toutes les expressions utilisées dans une spécification de métrique doivent finalement retourner une seule séries temporelles.

Pour utiliser les mathématiques de métrique, procédez comme suit :

- Choisissez un ou plusieurs CloudWatch indicateurs. Créez ensuite l'expression. Pour plus d'informations, consultez la section Utilisation des mathématiques métriques dans le guide de CloudWatch l'utilisateur Amazon.
- Vérifiez que l'expression mathématique de la métrique est valide à l'aide de la CloudWatch console ou de l' CloudWatchGetMetricDataAPI.

Exemple de politique de mise à l'échelle prédictive qui combine des métriques à l'aide des mathématiques de métriques (AWS CLI)

Parfois, au lieu de spécifier la métrique directement, vous devrez d'abord traiter ses données d'une certaine manière. Par exemple, une application peut extraire le travail d'une file d'attente Amazon SQS et vous souhaitez utiliser le nombre d'éléments dans la file d'attente comme critère de mise à l'échelle prédictive. Le nombre de messages dans la file d'attente ne définit pas uniquement le nombre d'instances dont vous avez besoin. Par conséquent, un travail supplémentaire est nécessaire pour créer une métrique qui peut être utilisée pour calculer le backlog par instance.

Ce qui suit est un exemple de politique de mise à l'échelle prédictive pour ce scénario. Il spécifie les métriques de mise à l'échelle et de charge qui sont basées sur la métrique `ApproximateNumberOfMessagesVisible` d'Amazon SQS, qui est le nombre de messages disponibles pour la récupération de la file d'attente. Il utilise également la `GroupInServiceInstances` métrique Amazon EC2 Auto Scaling et une expression mathématique pour calculer le backlog par instance pour la métrique de dimensionnement.

```
aws application-autoscaling put-scaling-policy --policy-name my-sqs-custom-metrics-
policy \
  --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
  --service-namespace ecs \
  --resource-id service/MyCluster/test \
  "MetricSpecifications": [
    {
      "TargetValue": 100,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Label": "Get the queue size (the number of messages waiting to be
 processed)",
            "Id": "queue_size",
            "MetricStat": {
              "Metric": {
                "MetricName": "ApproximateNumberOfMessagesVisible",
                "Namespace": "AWS/SQS",
                "Dimensions": [
                  {
                    "Name": "QueueName",
                    "Value": "my-queue"
                  }
                ]
```

```
            },
            "Stat": "Sum"
          },
          "ReturnData": false
        },
        {
          "Label": "Get the group size (the number of running instances)",
          "Id": "running_capacity",
          "MetricStat": {
            "Metric": {
              "MetricName": "GroupInServiceInstances",
              "Namespace": "AWS/AutoScaling",
              "Dimensions": [
                {
                  "Name": "AutoScalingGroupName",
                  "Value": "my-asg"
                }
              ]
            },
            "Stat": "Sum"
          },
          "ReturnData": false
        },
        {
          "Label": "Calculate the backlog per instance",
          "Id": "scaling_metric",
          "Expression": "queue_size / running_capacity",
          "ReturnData": true
        }
      ]
    },
    "CustomizedLoadMetricSpecification": {
      "MetricDataQueries": [
        {
          "Id": "load_metric",
          "MetricStat": {
            "Metric": {
              "MetricName": "ApproximateNumberOfMessagesVisible",
              "Namespace": "AWS/SQS",
              "Dimensions": [
                {
                  "Name": "QueueName",
                  "Value": "my-queue"
                }
```

```
                ],
              },
              "Stat": "Sum"
            },
            "ReturnData": true
          }
        ]
      }
    }
  ]
}
```

L'exemple renvoie l'ARN de la politique.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-sqs-custom-metrics-policy",
  "Alarms": []
}
```

Interconnectez les services Amazon ECS

Les applications qui s'exécutent dans le cadre de tâches Amazon ECS ont souvent besoin de recevoir des connexions depuis Internet ou de se connecter à d'autres applications exécutées dans les services Amazon ECS. Si vous avez besoin de connexions externes depuis Internet, nous vous recommandons d'utiliser Elastic Load Balancing. Pour plus d'informations sur l'équilibrage de charge intégrée, consultez the section called “Utiliser l'équilibrage de charge pour répartir le trafic de service”.

Si vous avez besoin d'une application pour vous connecter à d'autres applications qui s'exécutent dans les services Amazon ECS, Amazon ECS propose les méthodes suivantes pour le faire sans équilibreur de charge :

- Amazon ECS Service Connect

  Nous recommandons Service Connect, qui fournit une configuration Amazon ECS pour la découverte de services, la connectivité et la surveillance du trafic. Avec Service Connect, vos applications peuvent utiliser des noms abrégés et des ports standard pour se connecter aux services Amazon ECS du même cluster ou à d'autres clusters, y compris entre VPCs ceux-ci Région AWS.

Lorsque vous utilisez Service Connect, Amazon ECS gère toutes les étapes de la découverte de services : création des noms susceptibles d'être découverts, gestion dynamique des entrées pour chaque tâche au démarrage et à l'arrêt des tâches, exécution d'un agent dans chaque tâche configuré pour découvrir les noms. Votre application peut rechercher les noms en utilisant les fonctionnalités standard pour les noms DNS et en établissant des connexions. Si votre application le fait déjà, vous n'avez pas besoin de la modifier pour utiliser Service Connect.

Vous fournissez la configuration complète dans chaque définition de service et de tâche. Amazon ECS gère les modifications apportées à cette configuration lors de chaque déploiement de service, afin de garantir que toutes les tâches d'un déploiement se comportent de la même manière. Par exemple, l'un des problèmes courants liés à la découverte du DNS en tant que service est le contrôle d'une migration. Si vous modifiez un nom DNS pour qu'il pointe vers les nouvelles adresses IP de remplacement, le délai TTL maximal peut être nécessaire avant que tous les clients commencent à utiliser le nouveau service. Avec Service Connect, le déploiement du client met à jour la configuration en remplaçant les tâches du client. Vous pouvez configurer le disjoncteur de déploiement et toute autre configuration de déploiement pour affecter les modifications de Service Connect de la même manière que pour tout autre déploiement.

Pour de plus amples informations, veuillez consulter Utilisez Service Connect pour connecter les services Amazon ECS avec des noms abrégés.

- Découverte du service Amazon ECS

  Une autre approche de service-to-service communication est la communication directe à l'aide de la découverte de services. Dans cette approche, vous pouvez utiliser l'intégration de la découverte de AWS Cloud Map services avec Amazon ECS. À l'aide de la découverte de services, Amazon ECS synchronise la liste des tâches lancées avec AWS Cloud Map, ce qui permet de conserver un nom d'hôte DNS qui correspond aux adresses IP internes d'une ou de plusieurs tâches provenant de ce service en particulier. Les autres services d'Amazon VPC peuvent utiliser ce nom d'hôte DNS pour envoyer du trafic directement vers un autre conteneur à l'aide de son adresse IP interne.

  Cette approche de service-to-service communication permet une faible latence. Il n'y a aucun composant supplémentaire entre les récipients. Le trafic se déplace directement d'un conteneur à l'autre.

  Cette approche convient lorsque vous utilisez le mode `awsvpc` réseau, où chaque tâche possède sa propre adresse IP unique. La plupart des logiciels ne prennent en charge que l'utilisation d'Aenregistrements DNS, qui se résolvent directement en adresses IP. Lorsque vous utilisez le

mode `awsvpc` réseau, l'adresse IP de chaque tâche est un `A` enregistrement. Toutefois, si vous utilisez le mode `bridge` réseau, plusieurs conteneurs peuvent partager la même adresse IP. En outre, les mappages de ports dynamiques entraînent l'attribution aléatoire de numéros de port aux conteneurs sur cette adresse IP unique. À ce stade, un `A` enregistrement n'est plus suffisant pour la découverte de services. Vous devez également utiliser un `SRV` enregistrement. Ce type d'enregistrement permet de suivre à la fois les adresses IP et les numéros de port, mais nécessite que vous configuriez les applications de manière appropriée. Certaines applications prédéfinies que vous utilisez peuvent ne pas prendre en charge `SRV` les enregistrements.

Un autre avantage du mode `awsvpc` réseau est que vous disposez d'un groupe de sécurité unique pour chaque service. Vous pouvez configurer ce groupe de sécurité pour autoriser les connexions entrantes provenant uniquement des services en amont spécifiques qui doivent communiquer avec ce service.

Le principal inconvénient de la service-to-service communication directe utilisant la découverte de services est que vous devez implémenter une logique supplémentaire pour effectuer de nouvelles tentatives et faire face aux échecs de connexion. Les enregistrements DNS ont une période time-to-live (TTL) qui contrôle la durée pendant laquelle ils sont mis en cache. Il faut un certain temps pour que l'enregistrement DNS soit mis à jour et que le cache expire afin que vos applications puissent récupérer la dernière version de l'enregistrement DNS. Il se peut donc que votre application finisse par résoudre l'enregistrement DNS pour qu'il pointe vers un autre conteneur qui n'existe plus. Votre application doit gérer les nouvelles tentatives et disposer d'une logique lui permettant d'ignorer les mauvais backends.

Pour de plus amples informations, consultez Utilisez la découverte des services pour connecter les services Amazon ECS aux noms DNS.

- Réseau Amazon VPC

  Amazon VPC Lattice est un service de mise en réseau d'applications géré que les clients d'Amazon ECS utilisent pour observer, sécuriser et surveiller les applications créées à travers des services AWS informatiques et des comptes sans avoir à modifier leur code. VPCs

  VPC Lattice utilise des groupes cibles, qui sont un ensemble de ressources de calcul. Ces cibles exécutent votre application ou votre service et peuvent être des EC2 instances Amazon, des adresses IP, des fonctions Lambda et des équilibreurs de charge d'application. En associant leurs services Amazon ECS à un groupe cible VPC Lattice, les clients peuvent désormais activer les tâches Amazon ECS en tant que cibles IP dans VPC Lattice. Amazon ECS enregistre

automatiquement les tâches auprès du groupe cible VPC Lattice lorsque les tâches du service enregistré sont lancées.

Pour de plus amples informations, veuillez consulter Utilisez Amazon VPC Lattice pour connecter, observer et sécuriser vos services Amazon ECS.

Tableau de compatibilité des modes réseau

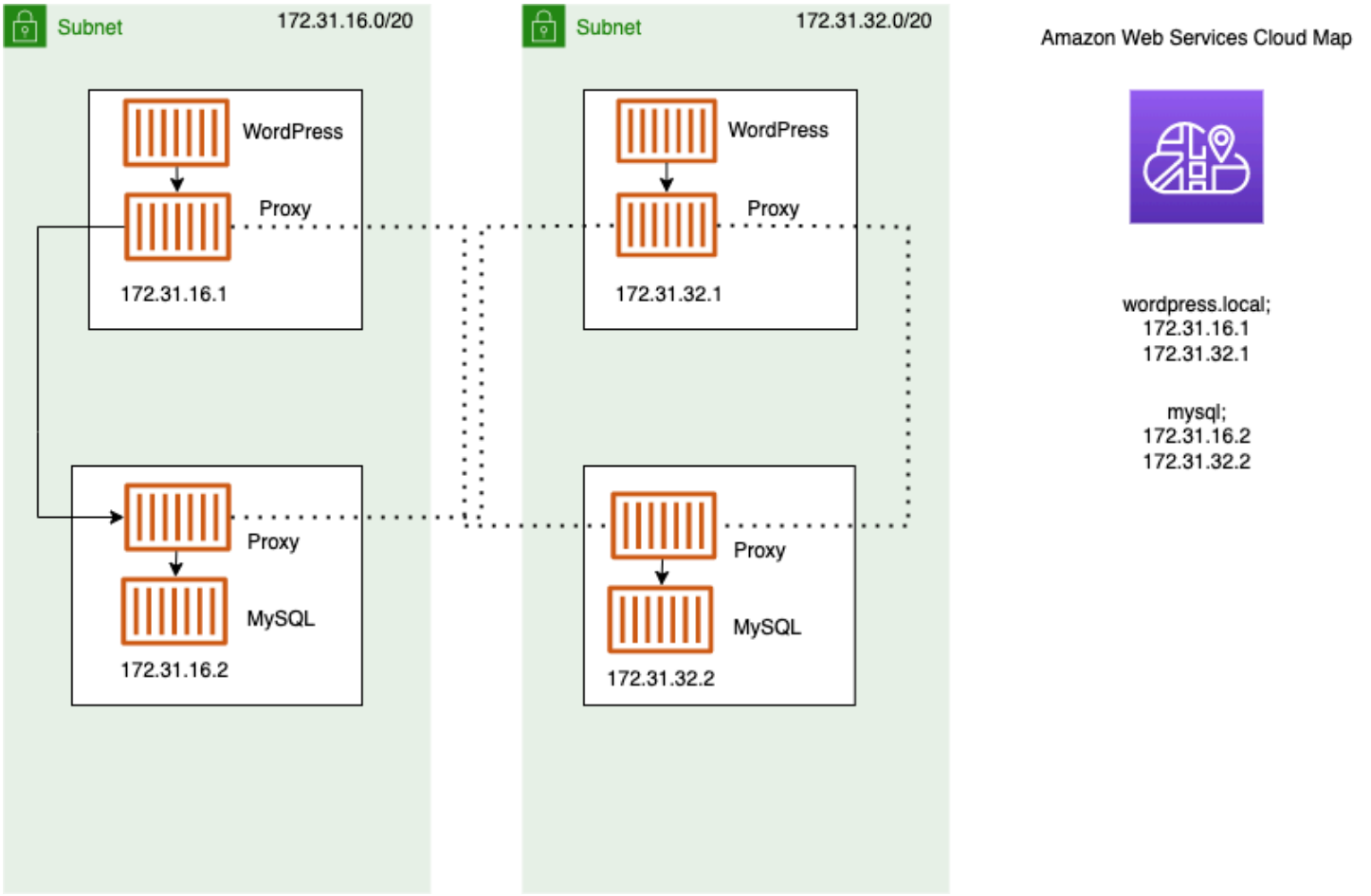
Le tableau suivant décrit la compatibilité entre ces options et les modes réseau des tâches. Dans le tableau, le terme « client » fait référence à l'application qui établit les connexions depuis une tâche Amazon ECS.

Options d'interconnexion	Pontée	**`awsvpc`**	Host (Hôte)
Découverte de service	oui, mais nécessite que les clients soient au courant des enregistrements SRV dans le DNS sans `hostPort`.	oui	oui, mais nécessite que les clients soient au courant des enregistrements SRV dans le DNS sans `hostPort`.
Service Connect	oui	oui	non
VPC Lattice	oui	oui	oui

Utilisez Service Connect pour connecter les services Amazon ECS avec des noms abrégés

Amazon ECS Service Connect permet de gérer les service-to-service communications en tant que configuration Amazon ECS. Il crée à la fois la découverte de services et un maillage de services dans Amazon ECS. Cela fournit la configuration complète de chaque service que vous gérez par déploiements de services, une manière unifiée de faire référence à vos services dans des espaces de noms qui ne dépendent pas de la configuration DNS du VPC, ainsi que des métriques et des journaux standardisés pour surveiller toutes vos applications. Service Connect interconnecte uniquement les services.

Le schéma suivant montre un exemple de réseau Service Connect avec deux sous-réseaux dans le VPC et deux services. Un service client qui s'exécute WordPress avec une tâche dans chaque sous-réseau. Un service de serveur qui exécute MySQL avec une tâche dans chaque sous-réseau. Les deux services sont hautement disponibles et résilients aux problèmes liés aux tâches et aux zones de disponibilité, car chaque service exécute plusieurs tâches réparties sur deux sous-réseaux. Les flèches continues indiquent une connexion depuis WordPress MySQL. Par exemple, une commande `mysql --host=mysql` CLI exécutée depuis l'intérieur du WordPress conteneur dans la tâche avec l'adresse IP`172.31.16.1`. La commande utilise le nom abrégé `mysql` du port par défaut de MySQL. Ce nom et ce port se connectent au proxy Service Connect dans le cadre de la même tâche. Le proxy utilisé dans la WordPress tâche utilise l'équilibrage de charge circulaire et toutes les informations relatives aux défaillances précédentes pour détecter les valeurs aberrantes afin de sélectionner la tâche MySQL à laquelle se connecter. Comme le montrent les flèches pleines du schéma, le proxy se connecte au deuxième proxy de la tâche MySQL avec l'adresse IP `172.31.16.2`. Le second proxy se connecte au serveur MySQL local dans le cadre de la même tâche. Les deux proxys indiquent les performances de connexion qui sont visibles dans les graphiques des CloudWatch consoles Amazon ECS et Amazon afin que vous puissiez obtenir les mesures de performance de toutes sortes d'applications de la même manière.



Les services suivants sont utilisés avec Service Connect.

nom du port

Configuration de définition de tâche Amazon ECS qui attribue un nom à un mappage de port particulier. Cette configuration est uniquement utilisée par Amazon ECS Service Connect.

alias du client

Configuration du service Amazon ECS qui attribue le numéro de port utilisé dans le point de terminaison. L'alias du client peut également attribuer le nom DNS du point de terminaison, en remplaçant le nom de découverte. Si aucun nom de découverte n'est fourni dans le service Amazon ECS, le nom d'alias du client remplace le nom du port comme nom de point de terminaison. Pour des exemples de points de terminaison, consultez la définition de point de terminaison. Plusieurs alias client peuvent être attribués à un service Amazon ECS. Cette configuration est uniquement utilisée par Amazon ECS Service Connect.

nom de découverte

Nom intermédiaire facultatif que vous pouvez créer pour un port spécifié à partir de la définition de tâche. Ce nom est utilisé pour créer un AWS Cloud Map service. Si ce nom n'est pas fourni, le nom de port indiqué dans la définition de tâche est utilisé. Plusieurs noms de découverte peuvent être attribués à un port spécifique d'un service Amazon ECS. Cette configuration est uniquement utilisée par Amazon ECS Service Connect.

AWS Cloud Map les noms de service doivent être uniques au sein d'un espace de noms. En raison de cette limitation, vous ne pouvez avoir qu'une seule configuration Service Connect sans nom de découverte pour une définition de tâche particulière dans chaque espace de noms.

point de terminaison

URL permettant de se connecter à une API ou à un site Web. L'URL contient le protocole, un nom DNS et le port. Pour plus d'informations sur les points de terminaison en général, veuillez consulter point de terminaison dans le glossaire AWS de la Référence générale d'Amazon Web Services(langue française non garantie).

Service Connect crée des points de terminaison qui se connectent aux services Amazon ECS et configure les tâches des services Amazon ECS pour se connecter aux points de terminaison. L'URL contient le protocole, un nom DNS et le port. Vous sélectionnez le protocole et le nom du port dans la définition de la tâche, car le port doit correspondre à l'application qui se trouve dans l'image du conteneur. Dans le service, vous sélectionnez chaque port par son nom et vous pouvez attribuer le nom DNS. Si vous ne spécifiez pas de nom DNS dans la configuration du service Amazon ECS, le nom de port indiqué dans la définition de tâche est utilisé par défaut. Par exemple, un point de terminaison Service Connect pourrait être `http://blog:80`, `grpc://checkout:8080` ou `http://_db.production.internal:99`.

Service Service Connect

Configuration d'un point de terminaison unique dans un service Amazon ECS. Cela fait partie de la configuration de Service Connect, qui se compose d'une seule ligne dans la configuration de Service Connect et du nom de découverte dans la console, ou d'un objet dans la liste `services` de la configuration JSON d'un service Amazon ECS. Cette configuration est uniquement utilisée par Amazon ECS Service Connect.

Pour plus d'informations, consultez le ServiceConnectServicemanuel Amazon Elastic Container Service API Reference.

espace de nom

Nom abrégé ou Amazon Resource Name (ARN) complet de l'espace de AWS Cloud Map noms à utiliser avec Service Connect. L'espace de noms doit être Région AWS identique à celui du service et du cluster Amazon ECS. Le type d'espace de noms dans AWS Cloud Map n'a aucune incidence sur Service Connect.

Service Connect utilise l' AWS Cloud Map espace de noms comme un regroupement logique de tâches Amazon ECS qui communiquent entre elles. Chaque service Amazon ECS ne peut appartenir qu'à un seul espace de noms. Les services d'un espace de noms peuvent être répartis sur différents clusters Amazon ECS au sein d'un même Région AWS espace. Compte AWS Vous pouvez librement organiser les services selon tous les critères.

service client

Service qui exécute une application cliente réseau. Un espace de noms doit être configuré pour ce service. Chaque tâche du service peut découvrir et se connecter à tous les points de terminaison de l'espace de noms via un conteneur de proxy Service Connect.

Si l'un des conteneurs de la tâche doit se connecter à un point de terminaison à partir d'un service situé dans un espace de noms, choisissez un service client. Si une application frontend, un proxy inverse ou une application d'équilibreur de charge reçoit du trafic externe via d'autres méthodes comme Elastic Load Balancing, elle peut utiliser ce type de configuration Service Connect.

service client-serveur

Service Amazon ECS qui exécute une application service réseau ou web. Un espace de noms et au moins un point de terminaison doivent être configurés pour ce service. Chaque tâche du service est accessible à l'aide des points de terminaison. Le conteneur de proxy Service Connect écoute le nom et le port du point de terminaison pour diriger le trafic vers les conteneurs d'applications de la tâche.

Si l'un des conteneurs expose et écoute le trafic réseau sur un port, choisissez un service client-serveur. Ces applications n'ont pas besoin de se connecter à d'autres services client-serveur dans le même espace de noms, mais la configuration du client est nécessaire. Un backend, un intergiciel, un niveau métier ou la plupart des microservices peuvent utiliser ce type de configuration Service Connect. Si vous souhaitez qu'une application frontend, un proxy inverse ou une application d'équilibreur de charge reçoive du trafic provenant d'autres services configurés avec Service Connect dans le même espace de noms, ces services doivent utiliser ce type de configuration Service Connect.

La fonctionnalité Service Connect crée un réseau virtuel de services connexes. La même configuration de service peut être utilisée sur plusieurs espaces de noms différents pour exécuter des ensembles d'applications indépendants mais identiques. Service Connect définit le conteneur de proxy dans le service Amazon ECS. Ainsi, la même définition de tâche peut être utilisée pour exécuter des applications identiques dans différents espaces de noms avec des configurations Service Connect différentes. Chaque tâche effectuée par le service exécute un conteneur proxy dans la tâche.

Service Connect convient aux connexions entre les services Amazon ECS au sein du même espace de noms. Pour les applications suivantes, vous devez utiliser une méthode d'interconnexion supplémentaire pour vous connecter à un service Amazon ECS configuré avec Service Connect :

- Tâches configurées dans d'autres espaces de noms
- Tâches non configurées pour Service Connect
- Autres applications en dehors d'Amazon ECS

Ces applications peuvent se connecter via le proxy Service Connect mais ne peuvent pas résoudre les noms des points de terminaison Service Connect.

Pour que ces applications résolvent les adresses IP des tâches Amazon ECS, vous devez utiliser une autre méthode d'interconnexion.

Rubriques

- Tarification
- Composants Amazon ECS Service Connect
- Présentation de la configuration d'Amazon ECS Service Connect
- Chiffrez le trafic Amazon ECS Service Connect
- Configuration d'Amazon ECS Service Connect avec AWS CLI

Tarification

- La tarification d'Amazon ECS Service Connect varie selon que vous utilisez AWS Fargate ou non l' EC2 infrastructure Amazon pour héberger vos charges de travail conteneurisées. Lorsque vous utilisez Amazon ECS sur Amazon AWS Outposts, la tarification suit le même modèle que celui utilisé lorsque vous utilisez Amazon EC2 directement. Pour plus d'informations, consultez Tarification Amazon ECS.

- L'utilisation d'Amazon ECS Service Connect est gratuite.
- AWS Cloud Map l'utilisation est totalement gratuite lorsqu'elle est utilisée par Service Connect.
- Les clients paient pour les ressources de calcul utilisées par Amazon ECS Service Connect, notamment le vCPU et la mémoire. Comme l'agent Amazon ECS Service Connect s'exécute dans le cadre d'une tâche client, son exécution est gratuite. Les ressources des tâches sont partagées entre la charge de travail du client et l'agent Amazon ECS Service Connect.
- Lorsqu'ils utilisent la fonctionnalité de chiffrement du trafic Amazon ECS Service Connect avec AWS Private CA, les clients paient pour l'autorité de certification privée qu'ils créent et pour chaque certificat TLS émis. Pour plus de détails, consultez AWS Private Certificate Authority les tarifs. Pour estimer le coût mensuel des certificats TLS, les clients doivent connaître le nombre de services Amazon ECS sur lesquels le protocole TLS est activé, le multiplier par le coût du certificat, puis le multiplier par six. Comme Amazon ECS Service Connect effectue automatiquement une rotation des certificats TLS tous les cinq jours, six certificats sont émis par service Amazon ECS, par mois, en moyenne.

Composants Amazon ECS Service Connect

Lorsque vous utilisez Amazon ECS Service Connect, vous configurez chaque service Amazon ECS pour exécuter une application serveur qui reçoit des demandes réseau (service client-serveur) ou pour exécuter une application client qui effectue les demandes (service client).

Lorsque vous vous préparez à utiliser Service Connect, commencez par un service client-serveur. Vous pouvez ajouter une configuration Service Connect à un nouveau service ou à un service existant. Amazon ECS crée un point de terminaison Service Connect dans l'espace de noms. En outre, Amazon ECS crée un nouveau déploiement dans le service pour remplacer les tâches en cours d'exécution.

Les tâches existantes et les autres applications peuvent continuer à se connecter aux points de terminaison existants et aux applications externes. Si un service client-serveur ajoute des tâches par extension, les nouvelles connexions des clients seront équilibrées entre toutes les tâches. Si un service client-serveur est mis à jour, les nouvelles connexions des clients seront équilibrées entre les tâches de la nouvelle version.

Les tâches existantes ne peuvent pas être résolues et se connecter au nouveau point de terminaison. Seules les nouvelles tâches dotées d'une configuration Service Connect dans le même espace de noms et qui commencent à s'exécuter après ce déploiement peuvent être résolues et se connecter à ce point de terminaison.

Cela signifie que l'opérateur de l'application cliente détermine quand la configuration de son application change, même si l'opérateur de l'application serveur peut modifier sa configuration à tout moment. La liste des points de terminaison dans l'espace de noms peut changer chaque fois qu'un service de l'espace de noms est déployé. Les tâches existantes et les tâches de remplacement continuent de se comporter de la même manière qu'après le dernier déploiement.

Considérez les exemples suivants.

Tout d'abord, supposons que vous créez une application disponible sur Internet public dans un seul AWS CloudFormation modèle et une seule AWS CloudFormation pile. La découverte publique et l'accessibilité doivent être créées en dernier lieu AWS CloudFormation, y compris le service client frontal. Les services doivent être créés dans cet ordre afin d'éviter toute période pendant laquelle le service client frontend fonctionne et est accessible au public, alors que le backend ne l'est pas. Cela permet d'éliminer l'envoi de messages d'erreur au public pendant cette période. Dans AWS CloudFormation, vous devez utiliser le `dependsOn` pour indiquer AWS CloudFormation que plusieurs services Amazon ECS ne peuvent pas être créés en parallèle ou simultanément. Vous devez ajouter la valeur `dependsOn` au service client frontal pour chaque service backend client-serveur auquel les tâches clientes se connectent.

Ensuite, supposons qu'un service frontal existe sans configuration Service Connect. Les tâches se connectent à un service backend existant. Ajoutez d'abord une configuration Service Connect client-serveur au service backend, en utilisant le même nom dans le DNS ou `clientAlias` que celui utilisé par le frontend. Cela crée un nouveau déploiement, donc toutes les détections d'annulation du déploiement ou AWS Management Console AWS CLI, AWS SDKs et d'autres méthodes pour annuler et rétablir le service principal au déploiement et à la configuration précédents. Si vous êtes satisfait des performances et du comportement du service backend, ajoutez une configuration Service Connect client ou client-serveur au service frontal. Seules les tâches du nouveau déploiement utilisent le proxy Service Connect qui est ajouté à ces nouvelles tâches. Si vous rencontrez des problèmes avec cette configuration, vous pouvez revenir en arrière et revenir à votre configuration précédente en utilisant la détection de l'annulation du déploiement ou AWS Management Console d'autres méthodes pour restaurer et rétablir le service principal à son déploiement et à sa configuration précédents. AWS CLI AWS SDKs Si vous utilisez un autre système de découverte de services basé sur le DNS au lieu de Service Connect, toutes les applications frontales ou clientes commencent à utiliser de nouveaux points de terminaison et à modifier la configuration des points de terminaison après l'expiration du cache DNS local, ce qui prend généralement plusieurs heures.

Réseaux

Par défaut, le proxy Service Connect écoute le mappage des ports `containerPort` de définition des tâches. Les règles de votre groupe de sécurité doivent autoriser le trafic entrant (entrant) vers ce port en provenance des sous-réseaux sur lesquels les clients seront exécutés.

Même si vous définissez un numéro de port dans la configuration du service Service Connect, cela ne modifie pas le port du service client-serveur écouté par le proxy Service Connect. Lorsque vous définissez ce numéro de port, Amazon ECS modifie le port du point de terminaison auquel les services clients se connectent, sur le proxy Service Connect au sein de ces tâches. Le proxy du service client se connecte au proxy du service client-serveur à l'aide du `containerPort`.

Si vous souhaitez modifier le port sur lequel le proxy Service Connect écoute, modifiez le `ingressPortOverride` dans la configuration Service Connect du service client-serveur. Si vous modifiez ce numéro de port, vous devez autoriser le trafic entrant sur ce port qui est utilisé par le trafic vers ce service.

Le trafic que vos applications envoient aux services Amazon ECS configurés pour Service Connect nécessite qu'Amazon VPC et les sous-réseaux disposent de règles de table de routage et de règles d'ACL réseau qui autorisent les numéros de port `containerPort` et `ingressPortOverride` que vous utilisez.

Vous pouvez utiliser Service Connect pour envoyer du trafic entre les deux VPCs. Les mêmes exigences relatives aux règles de table de routage, au réseau ACLs et aux groupes de sécurité s'appliquent aux deux VPCs.

Par exemple, deux clusters créent des tâches différentes VPCs. Un service dans chaque cluster est configuré pour utiliser le même espace de noms. Les applications dans ces deux services peuvent résoudre tous les points de terminaison de l'espace de noms sans aucune configuration DNS VPC. Cependant, les proxys ne peuvent pas se connecter à moins que les tables de routage VPC, le VPC ou les tables de routage de sous-réseau et le réseau ACLs VPC n'autorisent le trafic sur les numéros de port et. `containerPort` `ingressPortOverride`

Pour les tâches utilisant le mode `bridge` réseau, vous devez créer un groupe de sécurité doté d'une règle entrante autorisant le trafic sur la plage de ports dynamiques supérieure. Attribuez ensuite le groupe de sécurité à toutes les EC2 instances du cluster Service Connect.

Proxy Service Connect

Si vous créez ou mettez à jour un service avec la configuration Service Connect, Amazon ECS ajoute un nouveau conteneur à chaque nouvelle tâche dès son démarrage. Ce modèle d'utilisation d'un

conteneur séparé est nommé `sidecar`. Ce conteneur n'est pas présent dans la définition de la tâche et vous ne pouvez pas le configurer. Amazon ECS gère la configuration du conteneur dans le service. Cela vous permet de réutiliser les mêmes définitions de tâches entre plusieurs services, espaces de noms et tâches sans Service Connect.

Ressource de proxy

- Pour les définitions de tâches, vous devez définir les paramètres du processeur et de la mémoire.

  Nous vous recommandons d'ajouter 256 unités de processeur supplémentaires et au moins 64 MiB de mémoire au processeur de vos tâches et à la mémoire pour le conteneur proxy Service Connect. Dans AWS Fargate, la quantité de mémoire la plus faible que vous pouvez définir est de 512 Mio de mémoire. Sur Amazon EC2, la mémoire de définition des tâches est requise.
- Pour le service, vous définissez la configuration du journal dans la configuration Service Connect.
- Si vous prévoyez que les tâches de ce service reçoivent plus de 500 demandes par seconde à leur charge maximale, nous vous recommandons d'ajouter 512 unités de processeur à votre processeur de tâches dans cette définition de tâche pour le conteneur de proxy Service Connect.
- Si vous prévoyez de créer plus de 100 services Service Connect dans l'espace de noms ou 2 000 tâches au total pour tous les services Amazon ECS dans l'espace de noms, nous vous recommandons d'ajouter 128 Mio de mémoire à votre mémoire de tâche pour le conteneur de proxy Service Connect. Vous devez le faire dans chaque définition de tâche utilisée par tous les services Amazon ECS dans l'espace de noms.

Configuration du proxy

Vos applications se connectent au proxy dans le conteneur sidecar dans le cadre de la même tâche dans laquelle se trouve l'application. Amazon ECS configure la tâche et les conteneurs de telle sorte que les applications se connectent au proxy uniquement lorsque l'application est connectée aux noms des points de terminaison dans le même espace de noms. Tout le reste du trafic n'utilise pas le proxy. L'autre trafic inclut les adresses IP du même VPC, les points de terminaison AWS de service et le trafic externe.

Equilibrage de charge

Service Connect configure le proxy pour qu'il utilise la stratégie alternée pour équilibrer la charge entre les tâches d'un point de terminaison Service Connect. Le proxy local présent dans la tâche d'où provient la connexion choisit l'une des tâches du service client-serveur qui fournit le point de terminaison.

Prenons l'exemple d'une tâche exécutée WordPress dans un service configuré en tant que service client dans un espace de noms appelé local. Il existe un autre service avec deux tâches qui exécutent la base de données MySQL. Ce service est configuré pour fournir un point de terminaison appelé `mysql` via Service Connect dans le même espace de noms. Dans WordPress cette tâche, l' WordPress application se connecte à la base de données en utilisant le nom du point de terminaison. Les connexions portant ce nom sont dirigées vers le proxy qui s'exécute dans un conteneur annexe dans le cadre de la même tâche. Le proxy peut ensuite se connecter à l'une ou l'autre des tâches MySQL en utilisant la stratégie alternée.

Stratégies d'équilibrage de charge : alternance

Détection des valeurs aberrantes

Cette fonctionnalité utilise les données dont dispose le proxy concernant les échecs de connexion précédents pour éviter d'envoyer de nouvelles connexions aux hôtes dont les connexions ont échoué. Service Connect configure la fonctionnalité de détection des valeurs aberrantes du proxy afin de fournir des surveillances de l'état passives.

Dans l'exemple précédent, le proxy peut se connecter à l'une des tâches MySQL. Si le proxy a établi plusieurs connexions à une tâche MySQL spécifique et que 5 connexions ou plus ont échoué au cours des 30 dernières secondes, le proxy évite cette tâche MySQL pendant 30 à 300 secondes.

Nouvelle tentative

Service Connect configure le proxy pour qu'il tente à nouveau la connexion qui passe par le proxy et échoue, et la deuxième tentative évite d'utiliser l'hôte de la connexion précédente. Cela garantit que chaque connexion via Service Connect n'échoue pas pour des raisons ponctuelles.

Nombre de nouvelles tentatives : 2

Expiration

Service Connect configure le proxy pour qu'il attende au maximum que vos applications client-serveur répondent. La valeur du délai d'expiration par défaut est de 15 secondes, mais elle peut être mise à jour.

Paramètres facultatifs :

IdleTimeout - Durée en secondes pendant laquelle une connexion reste active lorsqu'elle est inactive. Une valeur de `0` désactive. `idleTimeout`

La `idleTimeout` valeur par défaut pour`HTTP/HTTP2/GRPC`est de 5 minutes.

La durée `idleTimeout` par défaut `TCP` est de 1 heure.

perRequestTimeout- Le temps d'attente pour que l'amont réponde avec une réponse complète par demande. La valeur est `0` désactivée`perRequestTimeout`. Cela ne peut être défini que lorsque le conteneur `appProtocol` de l'application est`HTTP/HTTP2/GRPC`. La valeur par défaut est de 15 secondes.

> **Note**
>
> S'`idleTimeout`il est défini sur une durée inférieure à`perRequestTimeout`, la connexion se ferme lorsque le `idleTimeout` est atteint et non le`perRequestTimeout`.

Considérations

Lorsque vous utilisez Service Connect, tenez compte des points suivants :

- Les tâches exécutées dans Fargate doivent utiliser la version de la plateforme Fargate Linux ou une version `1.4.0` ultérieure pour utiliser Service Connect.
- La version de l'agent Amazon ECS sur l'instance de conteneur doit être `1.67.2` ou supérieure.
- Les instances de conteneur doivent exécuter la version `20230428` ou une version ultérieure d'AMI Amazon Linux 2023 optimisée pour Amazon ECS, ou la version `2.0.20221115` d'AMI Amazon Linux 2 optimisée pour Amazon ECS pour utiliser Service Connect. Ces versions disposent de l'agent Service Connect en plus de l'agent de conteneur Amazon ECS. Pour plus d'informations sur l'agent Service Connect, consultez Amazon ECS Service Connect Agent sur GitHub.
- Les instances de conteneur doivent disposer de l'autorisation `ecs:Poll` pour la ressource `arn:aws:ecs:`*`region`*`:`*`0123456789012`*`:task-set/`*`cluster`*`/*`. Si vous utilisez le `ecsInstanceRole`, vous n'avez pas besoin d'ajouter d'autorisations supplémentaires. La stratégie gérée par `AmazonEC2ContainerServiceforEC2Role` dispose de ces autorisations. Pour de plus amples informations, veuillez consulter Rôle IAM d'instance de conteneur Amazon ECS.
- Les tâches qui utilisent le mode `bridge` réseau et utilisent Service Connect ne prennent pas en charge le paramètre de définition du `hostname` conteneur.
- Les définitions de tâches doivent définir la limite de mémoire des tâches pour utiliser Service Connect. Pour de plus amples informations, veuillez consulter Proxy Service Connect.
- Les définitions de tâches qui définissent les limites de mémoire des conteneurs ne sont pas prises en charge.

Vous pouvez définir des limites de mémoire de conteneur sur vos conteneurs, mais vous devez définir la limite de mémoire des tâches sur un nombre supérieur à la somme des limites de mémoire des conteneurs. Le processeur et la mémoire supplémentaires inclus dans les limites de tâches que vous n'allouez pas dans les limites de conteneur sont utilisés par le conteneur de proxy Service Connect et les autres conteneurs qui ne définissent pas de limites de conteneur. Pour de plus amples informations, veuillez consulter Proxy Service Connect.

- Vous pouvez configurer Service Connect pour utiliser n'importe quel espace de AWS Cloud Map noms de la même région dans la même Compte AWS région.
- Chaque service ne peut appartenir qu'à un seul espace de noms.
- Seules les tâches créées par les services sont prises en charge.
- Tous les points de terminaison doivent être uniques dans un espace de noms.
- Tous les noms de découverte doivent être uniques dans un espace de noms.
- Vous devez redéployer les services existants avant que les applications puissent résoudre de nouveaux points de terminaison. Les nouveaux points de terminaison ajoutés à l'espace de noms après le déploiement le plus récent ne seront pas ajoutés à la configuration de la tâche. Pour de plus amples informations, veuillez consulter the section called “Composants de Service Connect”.
- Service Connect ne supprime pas les espaces de noms lorsque des clusters sont supprimés. Vous devez supprimer les espaces de noms dans. AWS Cloud Map
- Le trafic Application Load Balancer est acheminé par défaut via l'agent Service Connect en `awsvpc` mode réseau. Si vous souhaitez que le trafic non lié au service contourne l'agent Service Connect, utilisez le `ingressPortOverride` paramètre dans la configuration de votre service Service Connect.
- Service Connect avec TLS ne prend pas en charge le mode réseau Bridge. Seul le mode `awsvpc` réseau est pris en charge.

Service Connect ne prend pas en charge les fonctionnalités suivantes :

- Conteneurs Windows
- HTTP 1.0
- Tâches autonomes
- Services utilisant les types de déploiement blue/green alimenté par CodeDeploy et externe
- L'instance de conteneur `External` pour Amazon ECS Anywhere n'est pas prise en charge par Service Connect.

- PPv2

Présentation de la configuration d'Amazon ECS Service Connect

Lorsque vous utilisez Service Connect, vous devez configurer certains paramètres dans vos ressources.

Le tableau suivant décrit les paramètres de configuration des ressources Amazon ECS.

Emplacement des paramètres	Type d'application	Description	Obligatoire
Définition de tâche	Client	Aucune modification n'est disponible pour Service Connect dans les définitions des tâches du client.	N/A
Définition de tâche	Client-serveur	Les serveurs doivent ajouter des champs de `name` aux ports dans les `portMappings` des conteneurs. Pour de plus amples informations, consultez `portMappings`.	Oui
Définition de tâche	Client-serveur	Les serveurs peuvent éventuellement fournir un protocole d'application (par exemple, HTTP) pour recevoir des métriques spécifiques au protocole pour leurs applications serveur (par exemple, `HTTP 5xx`).	Non
Définition des services	Client	Les services clients doivent ajouter une `serviceConnectConfiguration` pour configurer l'espace de noms à rejoindre. Cet espace de noms doit contenir tous les services de serveur que ce service doit découvrir. Pour de plus amples informations, veuillez	Oui

Emplacement des paramètres	Type d'application	Description	Obligatoire
		consulter `serviceConnectConfiguration`.	
Définition des services	Client-serveur	Les services de serveur doivent ajouter une `serviceConnectConfiguration` pour configurer les noms DNS, les numéros de port et l'espace de noms à partir desquels le service est disponible. Pour de plus amples informations, veuillez consulter `serviceConnectConfiguration`.	Oui
Cluster	Client	Les clusters peuvent ajouter un espace de noms Service Connect par défaut. Les nouveaux services du cluster héritent de l'espace de noms lorsque Service Connect est configuré dans un service.	Non
Cluster	Client-serveur	Aucune modification n'est disponible pour Service Connect dans les clusters qui s'appliquent aux services de serveur. Les définitions de tâches de serveur et les services doivent définir la configuration correspondante.	N/A

Vue d'ensemble des étapes de configuration de Service Connect

Les étapes suivantes fournissent une vue d'ensemble de la configuration de Service Connect.

> **Important**
>
> - Service Connect crée AWS Cloud Map des services dans votre compte. La modification manuelle de ces AWS Cloud Map ressources par le biais d' registering/deregistering instances, la modification des attributs d'instance ou la suppression d'un service peuvent entraîner un comportement inattendu pour le trafic de votre application ou pour les déploiements ultérieurs.
> - Service Connect ne prend pas en charge les liens dans la définition des tâches.

1. Ajoutez des noms de port aux mappages de port dans vos définitions de tâches. Vous pouvez également identifier le protocole de couche 7 de l'application afin d'obtenir des métriques supplémentaires.
2. Créez un cluster avec un espace de AWS Cloud Map noms ou créez l'espace de noms séparément. Pour simplifier l'organisation, créez un cluster portant le nom que vous souhaitez pour l'espace de noms et spécifiez le même nom pour l'espace de noms. Dans ce cas, Amazon ECS crée un nouvel espace de noms HTTP avec la configuration nécessaire. Service Connect n'utilise ni ne crée de zones hébergées DNS dans Amazon Route 53.
3. Configurez les services pour créer des points de terminaison Service Connect dans l'espace de noms.
4. Déployez des services pour créer les points de terminaison. Amazon ECS ajoute un conteneur de proxy Service Connect à chaque tâche et crée les points de terminaison Service Connect dans AWS Cloud Map. Ce conteneur n'est pas configuré dans la définition de tâche, et cette dernière peut être réutilisée sans modification pour créer plusieurs services dans le même espace de noms ou dans plusieurs espaces de noms.
5. Déployez des applications clientes en tant que services pour vous connecter aux points de terminaison. Amazon ECS les connecte aux points de terminaison Service Connect par le proxy Service Connect dans chaque tâche.

   Les applications utilisent le proxy uniquement pour se connecter aux points de terminaison Service Connect. Il n'existe aucune configuration supplémentaire pour utiliser le proxy. Le proxy effectue un équilibrage de charge alterné, une détection des valeurs aberrantes et de nouvelles tentatives. Pour plus d'informations sur le proxy, veuillez consulter Proxy Service Connect.
6. Surveillez le trafic via le proxy Service Connect sur Amazon CloudWatch.

Configuration de cluster

Vous pouvez définir un espace de noms par défaut pour Service Connect lorsque vous créez ou mettez à jour le cluster. Si vous spécifiez un nom d'espace de noms qui n'existe pas dans la même Région AWS et le même compte, un nouvel espace de noms HTTP est créé.

Si vous créez un cluster et que vous spécifiez un espace de noms Service Connect par défaut, le cluster attend dans l'état `PROVISIONING` pendant qu'Amazon ECS crée l'espace de noms. Vous pouvez voir un `attachment` dans l'état du cluster qui indique l'état de l'espace de noms. Les pièces jointes ne sont pas affichées par défaut dans le AWS CLI, vous devez les ajouter `--include ATTACHMENTS` pour les voir.

Configuration du service

Service Connect est conçu pour nécessiter une configuration minimale. Vous devez définir un nom pour chaque mappage de port que vous souhaitez utiliser avec Service Connect dans la définition de tâche. Dans le service, vous devez activer Service Connect et sélectionner un espace de noms pour créer un service client. Pour créer un service client-serveur, vous devez ajouter une configuration de service Service Connect unique qui correspond au nom de l'un des mappages de port. Amazon ECS réutilise le numéro de port et le nom de port figurant dans la définition de tâche pour définir le service et le point de terminaison Service Connect. Pour remplacer ces valeurs, vous pouvez utiliser les autres paramètres Discovery (Découverte), DNS et Port dans la console ou `discoveryName` et `clientAliases`, respectivement, dans l'API Amazon ECS.

Chiffrez le trafic Amazon ECS Service Connect

Amazon ECS Service Connect prend en charge le chiffrement automatique du trafic avec des certificats TLS (Transport Layer Security) pour les services Amazon ECS. Lorsque vous pointez vos services Amazon ECS vers un AWS Private Certificate Authority (AWS Private CA), Amazon ECS fournit automatiquement des certificats TLS pour chiffrer le trafic entre vos services Amazon ECS Service Connect. Amazon ECS génère, fait pivoter et distribue les certificats TLS utilisés pour le chiffrement du trafic.

Le chiffrement automatique du trafic avec Service Connect utilise des fonctionnalités de chiffrement de pointe pour sécuriser vos communications interservices afin de répondre à vos exigences en matière de sécurité. Il prend en charge AWS Private Certificate Authority les certificats TLS avec `256-bit ECDSA` `2048-bit RSA` chiffrement. Vous avez également un contrôle total sur les certificats privés et les clés de signature pour vous aider à respecter les exigences de conformité. Par défaut, le protocole TLS 1.3 est pris en charge, mais pas le protocole TLS 1.0 à 1.2. Service Connect prend en charge le protocole TLS 1.3 avec les chiffrements suivants :

- `TLS_AES_128_GCM_SHA256`
- `TLS_AES_256_GCM_SHA384`
- `TLS_CHACHA20_POLY1305_SHA256`

> **Note**
>
> Pour utiliser TLS 1.3, vous devez l'activer sur l'écouteur de la cible.
> Seul le trafic entrant et sortant passant par l'agent Amazon ECS est chiffré.

Contrôles de santé de Service Connect et Application Load Balancer

Vous pouvez utiliser Service Connect avec les contrôles de santé d'Application Load Balancer et le chiffrement TLS 1.3.

Configuration de l'Application Load Balancer

Configurez l'Application Load Balancer avec les paramètres suivants :

- Configurez un écouteur TLS avec une politique de sécurité TLS 1.3 (telle que `ELBSecurityPolicy-TLS13 -1-2-2021-06`). Pour plus d'informations, consultez la section Politiques de sécurité de votre Application Load Balancer.
- Créez un groupe cible avec les paramètres suivants :
  - Définissez le protocole sur HTTPS
  - Attachez le groupe cible à l'écouteur TLS
  - Configurez le port de contrôle de santé pour qu'il corresponde au port de conteneur de votre service Service Connect

Configuration de Service Connect

Configurez un service avec les paramètres suivants :

- Configurez le service pour utiliser le mode `awsvpc` réseau, car le mode `bridge` réseau n'est pas pris en charge.
- Activez Service Connect pour le service.
- Configurez la configuration de l'équilibreur de charge avec les paramètres suivants :

- Spécifiez le groupe cible que vous avez configuré pour votre Application Load Balancer
- Définissez le port du conteneur pour qu'il corresponde au port du conteneur du service Service Connect TLS
- Évitez `ingressPortOverride` de configurer le service. Pour plus d'informations, consultez le ServiceConnectServicemanuel Amazon Elastic Container Service API Reference.

Considérations

Tenez compte des points suivants lorsque vous utilisez Application Load Balancer, TLS et Service Connect :

- Utilisez le mode `awsvpc` réseau plutôt que le mode `bridge` réseau pour les contrôles de santé HTTPS lorsque vous utilisez Service Connect avec le chiffrement TLS. Les contrôles de santé HTTP continueront de fonctionner avec `bridge` le mode.
- Configurez le port de vérification de l'état du groupe cible pour qu'il corresponde au port de conteneur du service Service Connect, et non au port HTTPS par défaut (443).

AWS Private Certificate Authority certificats et Service Connect

Vous devez avoir le rôle IAM dans l'infrastructure. Pour plus d'informations sur ce rôle, consultez le rôle IAM de l'infrastructure Amazon ECS.

AWS Private Certificate Authority modes pour Service Connect

AWS Private Certificate Authority peut fonctionner selon deux modes : usage général et de courte durée.

- Usage général - Certificats pouvant être configurés avec n'importe quelle date d'expiration.
- De courte durée - Certificats d'une validité maximale de sept jours.

Amazon ECS prend en charge les deux modes, mais nous vous recommandons d'utiliser des certificats de courte durée. Par défaut, les certificats sont renouvelés tous les cinq jours, et leur utilisation en mode éphémère permet de réaliser des économies importantes par rapport à un usage général.

Service Connect ne prend pas en charge la révocation des certificats et utilise plutôt des certificats de courte durée avec une rotation fréquente des certificats. Vous avez le droit de modifier la fréquence

de rotation, de désactiver ou de supprimer les secrets à l'aide de la rotation gérée dans Secrets Manager, mais cela peut avoir les conséquences suivantes.

- Fréquence de rotation plus courte - Une fréquence de rotation plus courte entraîne des coûts plus élevés en raison AWS Private CA de l'augmentation de la charge de travail liée à la rotation entre Secrets Manager et Auto Scaling. AWS KMS
- Fréquence de rotation plus longue - Les communications de vos applications échouent si la fréquence de rotation dépasse sept jours.
- Suppression du secret - La suppression du secret entraîne l'échec de la rotation et a un impact sur les communications des applications clientes.

En cas d'échec de votre rotation secrète, un `RotationFailed` événement est publié dans AWS CloudTrail. Vous pouvez également configurer une CloudWatch alarme pour`RotationFailed`.

> **Important**
>
> N'ajoutez pas de répliques de régions aux secrets. Cela empêche Amazon ECS de supprimer le secret, car Amazon ECS n'est pas autorisé à supprimer des régions de la réplication. Si vous avez déjà ajouté la réplication, exécutez la commande suivante.
>
> ```
> aws secretsmanager remove-regions-from-replication \
>  --secret-id SecretId \
>  --remove-replica-regions region-name
> ```

Autorités de certification subordonnées

Vous pouvez importer n'importe quel certificat AWS Private CA, root ou subordonné, dans Service Connect TLS afin de délivrer des certificats d'entité finale pour les services. L'émetteur indiqué est considéré comme le signataire et la source de confiance partout. Vous pouvez émettre des certificats d'entité finale pour différentes parties de votre application auprès de différents subordonnés CAs. Lorsque vous utilisez le AWS CLI, fournissez l'Amazon Resource Name (ARN) de l'autorité de certification pour établir la chaîne de confiance.

Autorités de certification locales

Pour utiliser votre autorité de certification locale, vous devez créer et configurer une autorité de certification subordonnée dans. AWS Private Certificate Authority Cela garantit que tous les certificats

TLS émis pour vos charges de travail Amazon ECS partagent la chaîne de confiance avec les charges de travail que vous exécutez sur site et sont en mesure de se connecter en toute sécurité.

> **Important**
>
> Ajoutez le tag requis `AmazonECSManaged : true` dans votre AWS Private CA.

Infrastructure en tant que code

Lorsque vous utilisez Service Connect TLS avec des outils d'infrastructure en tant que code (IaC), il est important de configurer correctement vos dépendances afin d'éviter des problèmes tels que le blocage des services. Votre AWS KMS clé, si elle est fournie, votre rôle IAM et vos AWS Private CA dépendances doivent être supprimés après votre service Amazon ECS.

Service Connect et Secrets Manager

Lorsque vous utilisez Amazon ECS Service Connect avec le chiffrement TLS, le service interagit avec Secrets Manager de la manière suivante :

Service Connect utilise le rôle d'infrastructure fourni pour créer des secrets dans Secrets Manager. Ces secrets sont utilisés pour stocker les clés privées associées à vos certificats TLS afin de chiffrer le trafic entre vos services Service Connect.

> **Warning**
>
> La création et la gestion automatiques de ces secrets par Service Connect rationalisent le processus de mise en œuvre du chiffrement TLS pour vos services. Cependant, il est important d'être conscient des implications potentielles en matière de sécurité. Les autres rôles IAM disposant d'un accès en lecture à Secrets Manager peuvent accéder à ces secrets créés automatiquement. Cela pourrait exposer du matériel cryptographique sensible à des parties non autorisées, si les contrôles d'accès ne sont pas correctement configurés.
> Pour atténuer ce risque, suivez les meilleures pratiques suivantes :
>
> - Gérez et limitez soigneusement l'accès à Secrets Manager, en particulier pour les secrets créés par Service Connect.
> - Auditez régulièrement les rôles IAM et leurs autorisations pour vous assurer que le principe du moindre privilège est respecté.

Lorsque vous accordez un accès en lecture à Secrets Manager, pensez à exclure les clés privées TLS créées par Service Connect. Vous pouvez le faire en utilisant une condition dans vos politiques IAM pour exclure les secrets ARNs qui correspondent au modèle :

```
"arn:aws:secretsmanager:::secret:ecs-sc!"
```

Exemple de politique IAM qui refuse l'`GetSecretValue`action à tous les secrets avec le `ecs-sc!` préfixe :

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*"
        }
    ]
}
```

> **Note**
>
> Il s'agit d'un exemple général qui devra peut-être être ajusté en fonction de votre cas d'utilisation spécifique et de la configuration de votre AWS compte. Testez toujours vos politiques IAM de manière approfondie pour vous assurer qu'elles fournissent l'accès prévu tout en préservant la sécurité.

En comprenant comment Service Connect interagit avec Secrets Manager, vous pouvez mieux gérer la sécurité de vos services Amazon ECS tout en tirant parti des avantages du chiffrement TLS automatique.

Service Connect et AWS Key Management Service

Vous pouvez l'utiliser AWS Key Management Servicepour chiffrer et déchiffrer vos ressources Service Connect. AWS KMS est un service géré par AWS lequel vous pouvez créer et gérer des clés cryptographiques qui protègent vos données.

Lorsque vous utilisez AWS KMS Service Connect, vous pouvez choisir d'utiliser une clé AWS détenue qui AWS gère pour vous ou de choisir une AWS KMS clé existante. Vous pouvez également créer une nouvelle AWS KMS clé à utiliser.

Fournir votre propre clé de chiffrement

Vous pouvez fournir vos propres éléments clés ou utiliser un magasin de clés externe via AWS Key Management Service Importer votre propre clé dans AWS KMS, puis spécifier le nom de ressource Amazon (ARN) de cette clé dans Amazon ECS Service Connect.

Voici un exemple de AWS KMS politique. Remplacez les `user input` valeurs par les vôtres.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "id",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/role-name"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey",
        "kms:GenerateDataKeyPair"
      ],
      "Resource": "*"
    }
  ]
}
```

Pour plus d'informations sur les politiques clés, consultez la section Création d'une politique clé dans le Guide du AWS Key Management Service développeur.

> **Note**
>
> Service Connect prend uniquement en charge les AWS KMS clés de chiffrement symétriques. Vous ne pouvez utiliser aucun autre type de AWS KMS clé pour chiffrer vos ressources Service Connect. Pour savoir si une AWS KMS clé est une clé de chiffrement symétrique, consultez la section Identifier les clés KMS asymétriques.

Pour plus d'informations sur les clés de chiffrement AWS Key Management Service symétriques, consultez la section AWS KMS Clés de chiffrement symétriques dans le manuel du AWS Key Management Service développeur.

Activation du protocole TLS pour Amazon ECS Service Connect

Vous activez le chiffrement du trafic lorsque vous créez ou mettez à jour un service Service Connect.

Pour activer le chiffrement du trafic pour un service dans un espace de noms existant à l'aide du AWS Management Console

1. Vous devez avoir le rôle IAM dans l'infrastructure. Pour plus d'informations sur ce rôle, consultez le rôle IAM de l'infrastructure Amazon ECS.
2. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
3. Dans le panneau de navigation, choisissez Namespaces (Espaces de noms).
4. Choisissez l'espace de noms avec le service pour lequel vous souhaitez activer le chiffrement du trafic.
5. Choisissez le service pour lequel vous souhaitez activer le chiffrement du trafic.
6. Choisissez Update Service dans le coin supérieur droit et faites défiler la page vers le bas jusqu'à la section Service Connect.
7. Choisissez Activer le chiffrement du trafic sous les informations de service pour activer le protocole TLS.
8. Pour le rôle TLS Service Connect, choisissez un rôle IAM d'infrastructure existant ou créez-en un nouveau.
9. Pour Autorité de certification Signer, choisissez une autorité de certification existante ou créez-en une nouvelle.

Pour plus d'informations, consultez AWS Private Certificate Authority certificats et Service Connect.

10. Pour Choisir une AWS KMS key, choisissez une clé AWS détenue et gérée ou vous pouvez choisir une autre clé. Vous pouvez également choisir d'en créer un nouveau.

Pour un exemple d'utilisation du AWS CLI pour configurer le protocole TLS pour votre service,Configuration d'Amazon ECS Service Connect avec AWS CLI.

Vérification de l'activation du protocole TLS pour Amazon ECS Service Connect

Service Connect initie le protocole TLS au niveau de l'agent Service Connect et y met fin au niveau de l'agent de destination. Par conséquent, le code de l'application ne détecte jamais les interactions TLS. Suivez les étapes ci-dessous pour vérifier que le protocole TLS est activé.

1. Incluez la `openssl` CLI dans l'image de votre application.
2. Activez ECS Exec sur vos services pour vous connecter à vos tâches via SSM. Vous pouvez également lancer une EC2 instance Amazon dans le même Amazon VPC que le service.
3. Récupérez l'adresse IP et le port d'une tâche auprès d'un service que vous souhaitez vérifier. Vous pouvez récupérer l'adresse IP de la tâche dans la AWS Cloud Map console. Les informations se trouvent sur la page des détails du service correspondant à l'espace de noms.
4. Connectez-vous à l'une de vos tâches `execute-command` comme dans l'exemple suivant. Vous pouvez également vous connecter à l' EC2 instance Amazon créée à l'étape 2.

```
$ aws ecs execute-command --cluster cluster-name \
    --task task-id  \
    --container container-name \
    --interactive \
    --command "/bin/sh"
```

> **Note**
>
> L'appel direct du nom DNS ne révèle pas le certificat.

5. Dans le shell connecté, utilisez la `openssl` CLI pour vérifier et afficher le certificat associé à la tâche.

   Exemple :

```
openssl s_client -connect 10.0.147.43:6379 < /dev/null 2> /dev/null \
| openssl x509 -noout -text
```

Exemple de réponse :

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            <serial-number>
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: <issuer>
        Validity
            Not Before: Jan 23 21:38:12 2024 GMT
            Not After : Jan 30 22:38:12 2024 GMT
        Subject: <subject>
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
                pub:
                    <pub>
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Subject Alternative Name:
                DNS:redis.yelb-cftc
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Authority Key Identifier:
                keyid:<key-id>

            X509v3 Subject Key Identifier:
                1D:<id>
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
    Signature Algorithm: ecdsa-with-SHA256
        <hash>
```

Configuration d'Amazon ECS Service Connect avec AWS CLI

Vous pouvez créer un service Amazon ECS pour une tâche Fargate qui utilise Service Connect avec le. AWS CLI

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir de AWS CLI SDKs, et de l'API Amazon ECS via les deux IPv4 et. IPv6 Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

Prérequis

Les conditions requises pour Service Connect sont les suivantes :

- Vérifiez que la dernière version de AWS CLI est installée et configurée. Pour plus d'informations, voir Installation ou mise à jour vers la dernière version du AWS CLI.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique Amazon ECS_ FullAccess IAM.
- Vous avez un VPC, un sous-réseau, une table de routage et un groupe de sécurité prêts à être utilisés. Pour de plus amples informations, veuillez consulter the section called “Créer un Virtual Private Cloud”.
- Vous avez un rôle d'exécution de tâches portant le nom `ecsTaskExecutionRole` et la politique gérée par `AmazonECSTaskExecutionRolePolicy` est associée au rôle. Ce rôle permet à Fargate d'écrire les journaux de l'application NGINX et les journaux du proxy Service Connect sur Amazon Logs. CloudWatch Pour de plus amples informations, veuillez consulter Création du rôle d'exécution de tâche.

Étape 1 : créer le cluster

Pour créer votre espace de noms et votre cluster Amazon ECS, effectuez les étapes suivantes.

Pour créer le cluster et l'espace de AWS Cloud Map noms Amazon ECS

1. Créez un cluster Amazon ECS nommé `tutorial` que vous allez utiliser. Le paramètre `--service-connect-defaults` définit l'espace de noms par défaut du cluster. Dans l'exemple

de sortie, aucun AWS Cloud Map espace de noms du même nom `service-connect` n'existe dans ce compte. L'espace de noms est donc créé par Amazon ECS. Région AWS L'espace de noms est créé dans AWS Cloud Map dans le compte, et il est visible avec tous les autres espaces de noms. Utilisez donc un nom descriptif.

```
aws ecs create-cluster --cluster-name tutorial --service-connect-defaults
 namespace=service-connect
```

Sortie :

```
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
        "clusterName": "tutorial",
        "serviceConnectDefaults": {
            "namespace": "arn:aws:servicediscovery:us-
west-2:123456789012:namespace/ns-EXAMPLE"
        },
        "status": "PROVISIONING",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [
            {
                "name": "containerInsights",
                "value": "disabled"
            }
        ],
        "capacityProviders": [],
        "defaultCapacityProviderStrategy": [],
        "attachments": [
            {
                "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                "type": "sc",
                "status": "ATTACHING",
                "details": []
            }
        ],
        "attachmentsStatus": "UPDATE_IN_PROGRESS"
```

```
    }
}
}
```

2. Vérifiez que le cluster est créé :

```
aws ecs describe-clusters --clusters tutorial
```

Sortie :

```
{
    "clusters": [
        {
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
            "clusterName": "tutorial",
            "serviceConnectDefaults": {
                "namespace": "arn:aws:servicediscovery:us-
west-2:123456789012:namespace/ns-EXAMPLE"
            },
            "status": "ACTIVE",
            "registeredContainerInstancesCount": 0,
            "runningTasksCount": 0,
            "pendingTasksCount": 0,
            "activeServicesCount": 0,
            "statistics": [],
            "tags": [],
            "settings": [],
            "capacityProviders": [],
            "defaultCapacityProviderStrategy": []
        }
    ],
    "failures": []
}
```

3. (Facultatif) Vérifiez que l'espace de noms est créé dans AWS Cloud Map. Vous pouvez utiliser la configuration AWS Management Console ou la AWS CLI configuration normale telle qu'elle a été créée dans AWS Cloud Map.

   Par exemple, utilisez l' AWS CLI :

```
aws servicediscovery get-namespace --id ns-EXAMPLE
```

Sortie :

```
{
    "Namespace": {
        "Id": "ns-EXAMPLE",
        "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
EXAMPLE",
        "Name": "service-connect",
        "Type": "HTTP",
        "Properties": {
            "DnsProperties": {
                "SOA": {}
            },
            "HttpProperties": {
                "HttpName": "service-connect"
            }
        },
        "CreateDate": 1661749852.422,
        "CreatorRequestId": "service-connect"
    }
}
```

Étape 2 : créer le service pour le serveur

La fonctionnalité Service Connect est destinée à interconnecter plusieurs applications sur Amazon ECS. Au moins l'une de ces applications doit fournir un service Web auquel se connecter. Au cours de cette étape, vous allez créer :

- La définition de tâche qui utilise l'image officielle non modifiée du conteneur NGINX et qui inclut la configuration de Service Connect.
- Définition du service Amazon ECS qui configure Service Connect pour fournir la découverte des services et un proxy de maillage de service pour le trafic vers ce service. La configuration réutilise l'espace de noms par défaut de la configuration du cluster afin de réduire la quantité de configuration de service que vous effectuez pour chaque service.
- Le service Amazon ECS. Il exécute une tâche à l'aide de la définition de tâche et insère un conteneur supplémentaire pour le proxy Service Connect. Le proxy écoute sur le port depuis le mappage de ports du conteneur dans la définition de la tâche. Dans une application cliente exécutée dans Amazon ECS, le proxy intégré à la tâche client écoute les connexions sortantes

vers le nom du port de définition de la tâche, le nom de découverte du service ou le nom d'alias du client de service, ainsi que le numéro de port provenant de l'alias client.

Pour créer le service Web avec Service Connect

1. Enregistrez une définition de tâche compatible avec Fargate et qui utilise le mode réseau `awsvpc`. Procédez comme suit :

   a. Créez un fichier nommé `service-connect-nginx.json` avec le contenu de la définition de tâche suivante.

      Cette définition de tâche configure Service Connect en ajoutant les paramètres de `name` et de `appProtocol` au mappage des ports. Le nom du port permet de mieux identifier ce port dans la configuration du service lorsque plusieurs ports sont utilisés. Le nom du port est également utilisé par défaut comme nom détectable à utiliser par d'autres applications de l'espace de noms.

      La définition de tâche contient le rôle IAM de tâche, car ECS Exec est activé sur le service.

      > **Important**
      >
      > Cette définition de tâche utilise un `logConfiguration` pour envoyer la sortie nginx depuis `stdout` et `stderr` vers Amazon Logs. CloudWatch Ce rôle d'exécution de tâches ne dispose pas des autorisations supplémentaires requises pour créer le groupe de CloudWatch journaux Logs. Créez le groupe de CloudWatch journaux dans Logs à l'aide du AWS Management Console ou AWS CLI. Si vous ne souhaitez pas envoyer les journaux nginx à Logs, vous pouvez CloudWatch supprimer le. `logConfiguration`
      > Remplacez l' Compte AWS identifiant dans le rôle d'exécution de la tâche par votre Compte AWS identifiant.

```
{
    "family": "service-connect-nginx",
    "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::123456789012:role/ecsTaskRole",
    "networkMode": "awsvpc",
```

```
    "containerDefinitions": [
        {
        "name": "webserver",
        "image": "public.ecr.aws/docker/library/nginx:latest",
        "cpu": 100,
        "portMappings": [
            {
                "name": "nginx",
                "containerPort": 80,
                "protocol": "tcp",
                "appProtocol": "http"
            }
        ],
        "essential": true,
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-nginx",
                "awslogs-region": "region",
                "awslogs-stream-prefix": "nginx"
            }
        }
        }
    ],
    "cpu": "256",
    "memory": "512"
}
```

b. Enregistrez la définition de tâche à l'aide du fichier `service-connect-nginx.json` :

```
aws ecs register-task-definition --cli-input-json file://service-connect-
nginx.json
```

2. Créez un service :

   a. Créez un fichier nommé `service-connect-nginx-service.json` avec le contenu du service Amazon ECS que vous créez. Cet exemple utilise la définition de tâche créée à l'étape précédente. Le paramètre `awsvpcConfiguration` est obligatoire, car l'exemple de définition de tâche utilise le mode réseau `awsvpc`.

   Lorsque vous créez le service ECS, spécifiez le type de lancement Fargate et la `LATEST` version de plateforme qui prend en charge Service Connect. Les `securityGroups` et

les `subnets` doivent appartenir à un VPC répondant aux exigences requises pour utiliser Amazon ECS. Vous pouvez obtenir le groupe de sécurité et le sous-réseau IDs depuis la console Amazon VPC.

Ce service configure Service Connect en ajoutant le paramètre `serviceConnectConfiguration`. L'espace de noms n'est pas obligatoire car un espace de noms par défaut est configuré pour le cluster. Les applications clientes exécutées dans ECS dans l'espace de noms se connectent à ce service en utilisant le `portName` et le port dans les `clientAliases`. Par exemple, ce service est accessible via `http://nginx:80/`, car nginx fournit une page de bienvenue à l'emplacement racine `/`. Les applications externes qui ne s'exécutent pas dans Amazon ECS ou qui ne se trouvent pas dans le même espace de noms peuvent accéder à cette application via le proxy Service Connect en utilisant l'adresse IP de la tâche et le numéro de port figurant dans la définition de la tâche. Pour votre `tls` configuration, ajoutez le certificat correspondant `arn` à votre rôle`awsPcaAuthorityArn`, à votre `kmsKey` rôle et à celui `roleArn` de votre rôle IAM.

Ce service utilise un `logConfiguration` pour envoyer la sortie du proxy Service Connect depuis `stdout` et `stderr` vers Amazon CloudWatch Logs. Ce rôle d'exécution de tâches ne dispose pas des autorisations supplémentaires requises pour créer le groupe de CloudWatch journaux Logs. Créez le groupe de CloudWatch journaux dans Logs à l'aide du AWS Management Console ou AWS CLI. Nous vous recommandons de créer ce groupe de journaux et de stocker les journaux du proxy dans CloudWatch Logs. Si vous ne souhaitez pas envoyer les journaux du proxy à CloudWatch Logs, vous pouvez supprimer le`logConfiguration`.

```
{
    "cluster": "tutorial",
    "deploymentConfiguration": {
        "maximumPercent": 200,
        "minimumHealthyPercent": 0
    },
    "deploymentController": {
        "type": "ECS"
    },
    "desiredCount": 1,
    "enableECSManagedTags": true,
    "enableExecuteCommand": true,
    "launchType": "FARGATE",
    "networkConfiguration": {
```

```
        "awsvpcConfiguration": {
            "assignPublicIp": "ENABLED",
            "securityGroups": [
                "sg-EXAMPLE"
            ],
            "subnets": [
                "subnet-EXAMPLE",
                "subnet-EXAMPLE",
                "subnet-EXAMPLE"
            ]
           }
    },
    "platformVersion": "LATEST",
    "propagateTags": "SERVICE",
    "serviceName": "service-connect-nginx-service",
    "serviceConnectConfiguration": {
        "enabled": true,
        "services": [
            {
                "portName": "nginx",
                "clientAliases": [
                    {
                        "port": 80
                    }
                ],
                "tls": {
                   "issuerCertificateAuthority": {
                      "awsPcaAuthorityArn": "certificateArn"
                   },
                   "kmsKey": "kmsKey",
                   "roleArn": "iamRoleArn"
                }
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "region",
                "awslogs-stream-prefix": "service-connect-proxy"
            }
        }
    },
    "taskDefinition": "service-connect-nginx"
```

```
}
```

b. Créez un service à l'aide du `service-connect-nginx-service.json` fichier :

```
aws ecs create-service --cluster tutorial --cli-input-json file://service-
connect-nginx-service.json
```

Sortie :

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/tutorial/
service-connect-nginx-service",
        "serviceName": "service-connect-nginx-service",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 1,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "FARGATE",
        "platformVersion": "LATEST",
        "platformFamily": "Linux",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
service-connect-nginx:1",
        "deploymentConfiguration": {
            "deploymentCircuitBreaker": {
                "enable": false,
                "rollback": false
            },
            "maximumPercent": 200,
            "minimumHealthyPercent": 0
        },
        "deployments": [
            {
                "id": "ecs-svc/3763308422771520962",
                "status": "PRIMARY",
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/service-connect-nginx:1",
                "desiredCount": 1,
```

```
                "pendingCount": 0,
                "runningCount": 0,
                "failedTasks": 0,
                "createdAt": 1661210032.602,
                "updatedAt": 1661210032.602,
                "launchType": "FARGATE",
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "assignPublicIp": "ENABLED",
                        "securityGroups": [
                            "sg-EXAMPLE"
                        ],
                        "subnets": [
                            "subnet-EXAMPLEf",
                            "subnet-EXAMPLE",
                            "subnet-EXAMPLE"
                        ]
                    }
                },
                "rolloutState": "IN_PROGRESS",
                "rolloutStateReason": "ECS deployment ecs-
svc/3763308422771520962 in progress.",
                "failedLaunchTaskCount": 0,
                "replacedTaskCount": 0,
                "serviceConnectConfiguration": {
                    "enabled": true,
                    "namespace": "service-connect",
                    "services": [
                        {
                            "portName": "nginx",
                            "clientAliases": [
                                {
                                    "port": 80
                                }
                            ]
                        }
                    ],
                    "logConfiguration": {
                        "logDriver": "awslogs",
                        "options": {
                            "awslogs-group": "/ecs/service-connect-proxy",
                            "awslogs-region": "us-west-2",
```

```
                            "awslogs-stream-prefix": "service-connect-proxy"
                        },
                        "secretOptions": []
                    }
                },
                "serviceConnectResources": [
                    {
                        "discoveryName": "nginx",
                        "discoveryArn": "arn:aws:servicediscovery:us-
west-2:123456789012:service/srv-EXAMPLE"
                    }
                ]
            }
        ],
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "version": 0,
        "events": [],
        "createdAt": 1661210032.602,
        "placementConstraints": [],
        "placementStrategy": [],
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "assignPublicIp": "ENABLED",
                "securityGroups": [
                    "sg-EXAMPLE"
                ],
                "subnets": [
                    "subnet-EXAMPLE",
                    "subnet-EXAMPLE",
                    "subnet-EXAMPLE"
                ]
            }
        },
        "schedulingStrategy": "REPLICA",
        "enableECSManagedTags": true,
        "propagateTags": "SERVICE",
        "enableExecuteCommand": true
    }
}
```

La `serviceConnectConfiguration` que vous avez fournie apparaît dans le premier déploiement de la sortie. Lorsque vous apportez des modifications au service ECS de manière à modifier des tâches, un nouveau déploiement est créé par Amazon ECS.

Étape 3 : Vérifier que vous pouvez vous connecter

Pour vérifier que Service Connect est configuré et fonctionne, procédez comme suit pour vous connecter au service Web à partir d'une application externe. Consultez ensuite les mesures supplémentaires dans CloudWatch le proxy Service Connect créé.

Pour vous connecter au service Web à partir d'une application externe

- Connectez-vous à l'adresse IP de la tâche et au port du conteneur à l'aide de l'adresse IP de la tâche

  Utilisez le AWS CLI pour obtenir l'ID de la tâche, en utilisant le`aws ecs list-tasks --cluster tutorial`.

  Si vos sous-réseaux et votre groupe de sécurité autorisent le trafic depuis l'Internet public sur le port défini par la tâche, vous pouvez vous connecter à l'adresse IP publique depuis votre ordinateur. L'adresse IP publique n'étant pas disponible dans `describe-tasks`, les étapes consistent à se rendre sur Amazon EC2 AWS Management Console ou AWS CLI à obtenir les détails de l'interface Elastic Network.

  Dans cet exemple, une EC2 instance Amazon du même VPC utilise l'adresse IP privée de la tâche. L'application est nginx, mais l'en-tête `server: envoy` indique que le proxy Service Connect est utilisé. Le proxy Service Connect écoute sur le port du conteneur depuis la définition de la tâche.

```
$ curl -v 10.0.19.50:80/
*   Trying 10.0.19.50:80...
* Connected to 10.0.19.50 (10.0.19.50) port 80 (#0)
> GET / HTTP/1.1
> Host: 10.0.19.50
> User-Agent: curl/7.79.1
> Accept: */*
>
* Mark bundle as not supporting multiuse
```

```
< HTTP/1.1 200 OK
< server: envoy
< date: Tue, 23 Aug 2022 03:53:06 GMT
< content-type: text/html
< content-length: 612
< last-modified: Tue, 16 Apr 2019 13:08:19 GMT
< etag: "5cb5d3c3-264"
< accept-ranges: bytes
< x-envoy-upstream-service-time: 0
<
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

Pour consulter les métriques de Service Connect

Le proxy Service Connect crée des métriques d'application (connexion HTTP HTTP2, gRPC ou TCP) dans des métriques. CloudWatch Lorsque vous utilisez la CloudWatch console, consultez les dimensions métriques supplémentaires de DiscoveryName, (DiscoveryName, ServiceName,

ClusterName) et (TargetDiscoveryName, TargetDiscoveryName ServiceName, ClusterName) dans l'espace de noms Amazon ECS. Pour plus d'informations sur ces statistiques et les dimensions, consultez la section Afficher les mesures disponibles dans le guide de l'utilisateur Amazon CloudWatch Logs.

Utilisez la découverte des services pour connecter les services Amazon ECS aux noms DNS

Vous pouvez éventuellement configurer votre Amazon ECS service afin d'utiliser la découverte de service Amazon ECS. La découverte de services utilise des actions d' AWS Cloud Map API pour gérer les espaces de noms HTTP et DNS pour vos services Amazon ECS. Pour plus d'informations, consultez Présentation de AWS Cloud Map dans le Manuel du développeur AWS Cloud Map .

La découverte de services est disponible dans les AWS régions suivantes :

Nom de la région	Région
USA Est (Virginie du Nord)	us-east-1
USA Est (Ohio)	us-east-2
USA Ouest (Californie du Nord)	us-west-1
US West (Oregon)	us-west-2
Afrique (Le Cap)	af-south-1
Asie-Pacifique (Hong Kong)	ap-east-1
Asie-Pacifique (Mumbai)	ap-south-1
Asie-Pacifique (Hyderabad)	ap-south-2
Asie-Pacifique (Tokyo)	ap-northeast-1
Asie-Pacifique (Séoul)	ap-northeast-2
Asie-Pacifique (Osaka)	ap-northeast-3
Asie-Pacifique (Singapour)	ap-southeast-1

Nom de la région	Région
Asie-Pacifique (Sydney)	ap-southeast-2
Asie-Pacifique (Jakarta)	ap-southeast-3
Asie-Pacifique (Melbourne)	ap-southeast-4
Canada (Centre)	ca-central-1
Canada Ouest (Calgary)	ca-west-1
Chine (Beijing)	cn-north-1
China (Ningxia)	cn-northwest-1
Europe (Francfort)	eu-central-1
Europe (Zurich)	eu-central-2
Europe (Irlande)	eu-west-1
Europe (Londres)	eu-west-2
Europe (Paris)	eu-west-3
Europe (Milan)	eu-south-1
Europe (Stockholm)	eu-north-1
Israël (Tel Aviv)	il-central-1
Europe (Espagne)	eu-south-2
Moyen-Orient (EAU)	me-central-1
Moyen-Orient (Bahreïn)	me-south-1
Amérique du Sud (São Paulo)	sa-east-1
AWS GovCloud (USA Est)	us-gov-east-1

Nom de la région	Région
AWS GovCloud (US-Ouest)	us-gov-west-1

Concepts associés à la découverte de service

La découverte de service se compose des éléments suivants :

- Espace de noms de découverte de services : groupe logique de services de découverte de services partageant le même nom de domaine, par exemple l'endroit où vous souhaitez acheminer le trafic. `example.com` Vous pouvez créer un espace de noms en appelant la `aws servicediscovery create-private-dns-namespace` commande ou dans la console Amazon ECS. Vous pouvez utiliser la commande `aws servicediscovery list-namespaces` pour afficher les informations récapitulatives concernant les espaces de noms créés par le compte actuel. Pour plus d'informations sur les commandes de découverte de services, consultez `create-private-dns-namespace` et consultez `list-namespaces` le Guide de AWS CLI référence AWS Cloud Map (de découverte de services).
- Service de découverte de service : présent dans l'espace de noms de la découverte de service, il se compose du nom du service et de la configuration DNS correspondant à l'espace de noms. Il fournit les principaux composants suivants :
  - Registre des services : vous permet de rechercher un service via des actions DNS ou AWS Cloud Map API et de récupérer un ou plusieurs points de terminaison disponibles pouvant être utilisés pour vous connecter au service.
- Instance de découverte de service : existe dans le service de découverte de service et comprend les attributs associés à chaque service Amazon ECS service dans le répertoire de services.
  - Attributs de l'instance : les métadonnées suivantes sont ajoutées en tant qu'attributs personnalisés pour chaque service Amazon ECS service qui est configuré pour utiliser la découverte de service :
    - **`AWS_INSTANCE_IPV4`**— Pour `A` mémoire, l' IPv4 adresse renvoyée par Route 53 en réponse aux requêtes DNS et AWS Cloud Map lorsqu'elle découvre les détails de l'instance, par exemple,`192.0.2.44`.
    - **`AWS_INSTANCE_PORT`** – La valeur de port associées au service de découverte de service.
    - **`AVAILABILITY_ZONE`** – La zone de disponibilité dans laquelle la tâche a été lancée. Pour les tâches utilisant le type de EC2 lancement, il s'agit de la zone de disponibilité dans laquelle se

trouve l'instance de conteneur. Pour les tâches qui utilisent le type de lancement Fargate, il s'agit de la zone de disponibilité dans laquelle l'interface réseau Elastic existe.

- **`REGION`** – La région dans laquelle la tâche existe.
- **`ECS_SERVICE_NAME`** – Le nom du service Amazon ECS service auquel la tâche appartient.
- **`ECS_CLUSTER_NAME`** – Le nom du cluster Amazon ECS auquel la tâche appartient.
- **`EC2_INSTANCE_ID`** – L'ID de l'instance de conteneur sur laquelle la tâche a été placée. Cet attribut personnalisé n'est pas ajouté si la tâche utilise le type de lancement Fargate.
- **`ECS_TASK_DEFINITION_FAMILY`** – La famille de définition de tâche que la tâche utilise.
- **`ECS_TASK_SET_EXTERNAL_ID`** – Si un ensemble de tâches est créé pour un déploiement externe et est associé à un registre de découverte de service, alors l'attribut `ECS_TASK_SET_EXTERNAL_ID` contiendra l'ID externe de l'ensemble de tâches.

- Surveillances de l'état Amazon ECS : Amazon ECS permet d'effectuer des surveillances d'état régulières au niveau du conteneur. Si un point de terminaison échoue à la surveillance de l'état, il est supprimé du routage DNS et marqué comme défectueux.

Remarques relatives à la découverte de service

Les informations suivantes doivent être prises en compte lors de l'utilisation de la découverte de service :

- La découverte de service est prise en charge pour les tâches sur Fargate qui utilisent la version de plateforme 1.1.0 ou supérieure. Pour de plus amples informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.
- Les services configurés pour utiliser la découverte de service sont limités à 1 000 tâches par service. Cela est dû à un quota de service Route 53.
- Le flux de travail créer un service dans la console Amazon ECS prend uniquement en charge l'enregistrement des services dans des espaces de noms DNS privés. Lorsqu'un espace de noms DNS AWS Cloud Map privé est créé, une zone hébergée privée Route 53 est créée automatiquement.
- Les attributs DNS du VPC doivent être configurés pour une résolution DNS réussie. Pour plus d'informations sur la configuration des attributs, consultez Attributs DNS dans votre VPC dans le Guide de l'utilisateur Amazon VPC.
- Les registres DNS créés pour un service de découverte de service sont toujours enregistrés avec l'adresse IP privée de la tâche et non pas l'adresse IP publique, même lorsque des espaces de noms publics sont utilisés.

- La découverte de service exige que les tâches spécifient le mode réseau `awsvpc`, `bridge` ou `host` (`none` n'est pas pris en charge).
- Si la définition de tâche du service utilise le mode réseau `awsvpc`, vous pouvez créer n'importe quelle combinaison de registres `A` ou SRV pour chaque tâche de service. Un port est obligatoire si vous utilisez des registres SRV.
- Si la définition de tâche du service utilise le mode réseau `bridge` ou `host`, un registre SRV est le seul type de registre DNS pris en charge. Créez un registre SRV pour chaque tâche de service. Le registre SRV doit spécifier une combinaison de nom de conteneur et de port de conteneur à partir de la définition de tâche.
- Vous pouvez interroger les registres DNS pour un service de découverte de service au sein de votre VPC. Ces enregistrements utilisent le format suivant : `<service-discovery-service-name>.<service-discovery-namespace>`.
- Lors de l'exécution d'une requête DNS sur le nom du service, les registres `A` renvoient un ensemble d'adresses IP correspondant à vos tâches. Les registres SRV renvoient un ensemble d'adresses IP et de ports pour chaque tâche.
- Si vous disposez de huit registres sains ou moins, Route 53 répond à toutes les requêtes DNS par tous les registres sains.
- Lorsque tous les registres sont non sains, Route 53 répond à toutes les requêtes DNS par jusqu'à huit registres non sains.
- Vous pouvez configurer la découverte de service pour un service situé derrière un équilibreur de charge, mais le trafic de découverte de service est toujours acheminé vers la tâche et non pas vers l'équilibreur de charge.
- La découverte de service ne prend pas en charge l'utilisation des équilibreurs de charge Classic Load Balancer.
- Nous vous recommandons d'utiliser les surveillances de l'état au niveau des conteneurs gérés par Amazon ECS pour votre service de découverte de service.
  - HealthCheckCustomConfig—Amazon ECS gère les bilans de santé en votre nom. Amazon ECS utilise les informations obtenues par le conteneur et les surveillances de l'état, et l'état de votre tâche, afin de mettre à jour l'état avec AWS Cloud Map. Cette configuration est spécifiée à l'aide du paramètre `--health-check-custom-config` au moment de la création de votre service de découverte de service. Pour plus d’informations, consultez HealthCheckCustomConfig dans la Référence d’API AWS Cloud Map .
- Les AWS Cloud Map ressources créées lors de l'utilisation de la découverte de services doivent être nettoyées manuellement.

- Les tâches et les instances sont enregistrées `UNHEALTHY` jusqu'à ce que les contrôles de santé du conteneur renvoient une valeur. Si les bilans de santé sont réussis, le statut est mis à jour à`HEALTHY`. Si les vérifications de l'état du conteneur échouent, l'instance de découverte de service est désenregistrée.

Tarification de la découverte de service

Les clients utilisant la découverte de service Amazon ECS service sont facturés pour les ressources Route 53 et les opérations d'API de découverte AWS Cloud Map . Le montant facturé englobe les coûts associés à la création des zones hébergées Route 53 et des requêtes sur le registre de services. Pour plus d'informations, consultez Tarification AWS Cloud Map dans le Guide du développeur AWS Cloud Map .

Amazon ECS effectue des contrôles de santé au niveau des conteneurs et les expose à des opérations d'API de vérification d'état AWS Cloud Map personnalisées. Ceci est actuellement mis à la disposition des clients sans frais supplémentaires. Si vous configurez des surveillances de l'état du réseau supplémentaires pour les tâches exposées publiquement, vous êtes facturé pour ces surveillances de l'état.

Création d'un service Amazon ECS utilisant Service Discovery

Découvrez comment créer un service contenant une tâche Fargate qui utilise la découverte de services avec le. AWS CLI

Pour obtenir la liste de Régions AWS ces services d'assistance découverts, consultezUtilisez la découverte des services pour connecter les services Amazon ECS aux noms DNS.

Pour de plus amples informations sur les Régions qui prennent en charge Fargate, veuillez consulter the section called “AWS Régions de Fargate”.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir de AWS CLI SDKs, et de l'API Amazon ECS via les deux IPv4 et. IPv6 Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

Prérequis

Avant de commencer ce didacticiel, vérifiez que vous respectez les conditions requises suivantes :

- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations, voir Installation ou mise à jour vers la dernière version du AWS CLI.
- Les étapes décrites dans Configurer l'utilisation d'Amazon ECS sont terminées.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique Amazon ECS_ FullAccess IAM.
- Vous avez créé au moins un VPC et un groupe de sécurité. Pour de plus amples informations, veuillez consulter the section called “Créer un Virtual Private Cloud”.

Étape 1 : créer les ressources Service Discovery dans AWS Cloud Map

Suivez ces étapes suivantes pour créer votre espace de noms de découverte de service et votre service de découverte de service :

1. Créez un espace de noms de découverte de service Cloud Map privé. Cet exemple crée un espace de noms appelé `tutorial`. *`vpc-abcd1234`*Remplacez-le par l'identifiant de l'un de vos identifiants existants VPCs.

   ```
   aws servicediscovery create-private-dns-namespace \
         --name tutorial \
         --vpc vpc-abcd1234
   ```

   La sortie de cette commande est la suivante.

   ```
   {
       "OperationId": "h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e"
   }
   ```

2. À l'aide du paramètre `OperationId` obtenu à partir de la sortie de l'étape précédente, vérifiez que l'espace de noms privés a bien été créé. Notez l'ID de l'espace de noms car vous l'utiliserez dans les commandes suivantes.

   ```
   aws servicediscovery get-operation \
         --operation-id h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e
   ```

   La sortie est la suivante.

```
{
    "Operation": {
        "Id": "h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e",
        "Type": "CREATE_NAMESPACE",
        "Status": "SUCCESS",
        "CreateDate": 1519777852.502,
        "UpdateDate": 1519777856.086,
        "Targets": {
           "NAMESPACE": "ns-uejictsjen2i4eeg"
        }
    }
}
```

3. En utilisant l'ID NAMESPACE indiqué dans la sortie de l'étape précédente, créez un service de découverte de service. Cet exemple crée un service nommé `myapplication`. Notez l'ID du service et l'ARN car vous les utiliserez dans les commandes suivantes.

```
aws servicediscovery create-service \
      --name myapplication \
      --dns-config "NamespaceId="ns-
uejictsjen2i4eeg",DnsRecords=[{Type="A",TTL="300"}]" \
      --health-check-custom-config FailureThreshold=1
```

La sortie est la suivante.

```
{
    "Service": {
       "Id": "srv-utcrh6wavdkggqtk",
        "Arn": "arn:aws:servicediscovery:region:aws_account_id:service/srv-
utcrh6wavdkggqtk",
        "Name": "myapplication",
        "DnsConfig": {
            "NamespaceId": "ns-uejictsjen2i4eeg",
            "DnsRecords": [
                {
                    "Type": "A",
                    "TTL": 300
                }
            ]
        },
        "HealthCheckCustomConfig": {
```

```
            "FailureThreshold": 1
        },
        "CreatorRequestId": "e49a8797-b735-481b-a657-b74d1d6734eb"
    }
}
```

Étape 2 : Créer les ressources Amazon ECS

Suivez ces étapes pour créer votre cluster, votre définition de tâche et votre service Amazon ECS service :

1. Créez un nouveau cluster Amazon ECS. Cet exemple crée un cluster appelé `tutorial`.

```
aws ecs create-cluster \
      --cluster-name tutorial
```

2. Enregistrez une définition de tâche compatible avec Fargate et qui utilise le mode réseau `awsvpc`. Procédez comme suit :

   a. Créez un fichier nommé `fargate-task.json` avec le contenu de la définition de tâche suivante.

```
{
    "family": "tutorial-task-def",
        "networkMode": "awsvpc",
        "containerDefinitions": [
            {
                "name": "sample-app",
                "image": "public.ecr.aws/docker/library/httpd:2.4",
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80,
                        "protocol": "tcp"
                    }
                ],
                "essential": true,
                "entryPoint": [
                    "sh",
                    "-c"
                ],
                "command": [
```

```
                    "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style>
 </head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample
 App</h1> <h2>Congratulations!</h2> <p>Your application is now running on a
 container in Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/
htdocs/index.html && httpd-foreground\""
                ]
            }
        ],
        "requiresCompatibilities": [
            "FARGATE"
        ],
        "cpu": "256",
        "memory": "512"
}
```

b. Enregistrez la définition de tâche en utilisant `fargate-task.json`.

```
aws ecs register-task-definition \
      --cli-input-json file://fargate-task.json
```

3. Créez un service ECS en procédant comme suit :

a. Créez un fichier nommé `ecs-service-discovery.json` avec le contenu du service ECS que vous créez. Cet exemple utilise la définition de tâche créée à l'étape précédente. Le paramètre `awsvpcConfiguration` est obligatoire, car l'exemple de définition de tâche utilise le mode réseau `awsvpc`.

Lorsque vous créez le service ECS, spécifiez le type de lancement Fargate et la version de plateforme `LATEST`, qui prend en charge la fonction de découverte de service. Lorsque le service de découverte de service est créé dans AWS Cloud Map , `registryArn` est l'ARN renvoyé. Les `securityGroups` et les `subnets` doivent appartenir au VPC utilisé pour créer l'espace de noms Cloud Map. Vous pouvez obtenir le groupe de sécurité et le sous-réseau IDs depuis la console Amazon VPC.

```
{
    "cluster": "tutorial",
    "serviceName": "ecs-service-discovery",
    "taskDefinition": "tutorial-task-def",
    "serviceRegistries": [
       {
```

```
            "registryArn":
  "arn:aws:servicediscovery:region:aws_account_id:service/srv-utcrh6wavdkggqtk"
        }
     ],
     "launchType": "FARGATE",
     "platformVersion": "LATEST",
     "networkConfiguration": {
        "awsvpcConfiguration": {
           "assignPublicIp": "ENABLED",
           "securityGroups": [ "sg-abcd1234" ],
           "subnets": [ "subnet-abcd1234" ]
        }
     },
     "desiredCount": 1
}
```

b. Créez votre service ECS à l'aide du `ecs-service-discovery.json`.

```
aws ecs create-service \
      --cli-input-json file://ecs-service-discovery.json
```

Étape 3 : vérifier la découverte du service dans AWS Cloud Map

Vous pouvez vérifier que tout est bien créé en interrogeant les informations associées à la fonction de découverte de service. Une fois la découverte des services configurée, vous pouvez soit utiliser des opérations d' AWS Cloud Map API, soit effectuer un appel `dig` depuis une instance au sein de votre VPC. Procédez comme suit :

1. À l'aide de l'ID de service de découverte de service, répertoriez les instances de découverte de service. Notez l'ID de l'instance (en gras) pour le nettoyage des ressources.

```
aws servicediscovery list-instances \
      --service-id srv-utcrh6wavdkggqtk
```

La sortie est la suivante.

```
{
    "Instances": [
        {
            "Id": "16becc26-8558-4af1-9fbd-f81be062a266",
            "Attributes": {
```

```
                "AWS_INSTANCE_IPV4": "172.31.87.2"
                "AWS_INSTANCE_PORT": "80",
                "AVAILABILITY_ZONE": "us-east-1a",
                "REGION": "us-east-1",
                "ECS_SERVICE_NAME": "ecs-service-discovery",
                "ECS_CLUSTER_NAME": "tutorial",
                "ECS_TASK_DEFINITION_FAMILY": "tutorial-task-def"
            }
        }
    ]
}
```

2. Utilisez l'espace de noms, le service et les paramètres supplémentaires tels que le nom du cluster ECS pour demander des informations détaillées sur les instances de la fonction de découverte de service.

```
aws servicediscovery discover-instances \
      --namespace-name tutorial \
      --service-name myapplication \
      --query-parameters ECS_CLUSTER_NAME=tutorial
```

3. Vous pouvez interroger les registres DNS créés dans la zone hébergée Route 53 de votre service de découverte de service à l'aide des commandes AWS CLI suivantes :

   a. À l'aide de l'ID de l'espace de noms, obtenez les informations relatives à l'espace de noms, lesquelles incluent l'ID de la zone hébergée Route 53.

```
aws servicediscovery \
      get-namespace --id ns-uejictsjen2i4eeg
```

   La sortie est la suivante.

```
{
    "Namespace": {
        "Id": "ns-uejictsjen2i4eeg",
        "Arn": "arn:aws:servicediscovery:region:aws_account_id:namespace/ns-
uejictsjen2i4eeg",
        "Name": "tutorial",
        "Type": "DNS_PRIVATE",
        "Properties": {
             "DnsProperties": {
                "HostedZoneId": "Z35JQ4ZFDRYPLV"
```

```
            }
        },
        "CreateDate": 1519777852.502,
        "CreatorRequestId": "9049a1d5-25e4-4115-8625-96dbda9a6093"
    }
}
```

b. À l'aide de l'ID de zone hébergée Route 53 de l'étape précédente (voir le texte en gras), obtenez l'enregistrement de ressource défini pour la zone hébergée.

```
aws route53 list-resource-record-sets \
      --hosted-zone-id Z35JQ4ZFDRYPLV
```

4. Vous pouvez également interroger le DNS à partir d'une instance de votre VPC à l'aide de `dig`.

```
dig +short myapplication.tutorial
```

Étape 4 : nettoyer

Une fois que vous avez terminé ce didacticiel, nettoyez les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources inutilisées. Procédez comme suit :

1. Annulez l'enregistrement des instances de service de découverte de service à l'aide de l'ID de service et de l'ID de l'instance que vous avez notés précédemment.

```
aws servicediscovery deregister-instance \
      --service-id srv-utcrh6wavdkggqtk \
      --instance-id 16becc26-8558-4af1-9fbd-f81be062a266
```

La sortie est la suivante.

```
{
    "OperationId": "xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv"
}
```

2. À l'aide du `OperationId` indiqué dans la sortie de l'étape précédente, vérifiez que l'inscription des instances du service de découverte de service ont été correctement annulées.

```
aws servicediscovery get-operation \
      --operation-id xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv
```

```
{
  "Operation": {
        "Id": "xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv",
        "Type": "DEREGISTER_INSTANCE",
        "Status": "SUCCESS",
        "CreateDate": 1525984073.707,
        "UpdateDate": 1525984076.426,
        "Targets": {
            "INSTANCE": "16becc26-8558-4af1-9fbd-f81be062a266",
            "ROUTE_53_CHANGE_ID": "C5NSRG1J4I1FH",
            "SERVICE": "srv-utcrh6wavdkggqtk"
        }
    }
}
```

3. Supprimez le service de découverte de service en utilisant l'ID du service.

```
aws servicediscovery delete-service \
      --id srv-utcrh6wavdkggqtk
```

4. Supprimer l'espace de noms de découverte de service en utilisant l'ID de l'espace de noms.

```
aws servicediscovery delete-namespace \
      --id ns-uejictsjen2i4eeg
```

La sortie est la suivante.

```
{
    "OperationId": "c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj"
}
```

5. À l'aide du `OperationId` indiqué dans la sortie de l'étape précédente, vérifiez que l'espace de noms de la découverte de service a été correctement supprimé.

```
aws servicediscovery get-operation \
      --operation-id c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj
```

La sortie est la suivante.

```
{
```

```
    "Operation": {
        "Id": "c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj",
        "Type": "DELETE_NAMESPACE",
        "Status": "SUCCESS",
        "CreateDate": 1525984602.211,
        "UpdateDate": 1525984602.558,
        "Targets": {
            "NAMESPACE": "ns-rymlehshst7hhukh",
            "ROUTE_53_CHANGE_ID": "CJP2A2M86XW3O"
        }
    }
}
```

6. Définissez le nombre souhaité pour le service Amazon ECS sur `0`. Vous devez effectuer cette étape pour supprimer le service à l'étape suivante.

```
aws ecs update-service \
      --cluster tutorial \
      --service ecs-service-discovery \
      --desired-count 0
```

7. Supprimez l'Amazon ECS service.

```
aws ecs delete-service \
      --cluster tutorial \
      --service ecs-service-discovery
```

8. Supprimez le cluster Amazon ECS.

```
aws ecs delete-cluster \
      --cluster tutorial
```

Utilisez Amazon VPC Lattice pour connecter, observer et sécuriser vos services Amazon ECS

Amazon VPC Lattice est un service de mise en réseau d'applications entièrement géré qui permet aux clients d'Amazon ECS d'observer, de sécuriser et de surveiller les applications basées sur des services AWS informatiques et des comptes VPCs, sans avoir à modifier le code.

VPC Lattice utilise des groupes cibles, qui sont un ensemble de ressources de calcul. Ces cibles exécutent votre application ou votre service et peuvent être des EC2 instances Amazon, des

adresses IP, des fonctions Lambda et des équilibreurs de charge d'application. En associant leurs services Amazon ECS à un groupe cible VPC Lattice, les clients peuvent désormais activer les tâches Amazon ECS en tant que cibles IP dans VPC Lattice. Amazon ECS enregistre automatiquement les tâches auprès du groupe cible VPC Lattice lorsque les tâches du service enregistré sont lancées.

> **Note**
>
> Lorsque vous utilisez cinq configurations VPC Lattice, votre temps de déploiement peut être légèrement plus long que lorsque vous utilisez moins de configurations.

Une règle d'écoute est utilisée pour transférer le trafic vers un groupe cible spécifié lorsque les conditions sont remplies. Un écouteur vérifie les demandes de connexion à l'aide du protocole du port que vous avez configuré. Un service achemine les demandes vers ses cibles enregistrées en fonction des règles que vous définissez lors de la configuration de votre écouteur.

Amazon ECS remplace également automatiquement une tâche si elle devient défectueuse selon les tests de santé de VPC Lattice. Une fois associés à VPC Lattice, les clients d'Amazon ECS peuvent également tirer parti des nombreuses autres fonctionnalités de connectivité, de sécurité et d'observabilité entre ordinateurs de VPC Lattice, telles que la connexion aux services entre clusters et comptes AWS Resource Access Manager avec VPCs, l'intégration IAM pour l'autorisation et l'authentification, et les fonctionnalités avancées de gestion du trafic.

Les clients Amazon ECS peuvent bénéficier de VPC Lattice des manières suivantes.

- Productivité accrue des développeurs - VPC Lattice améliore la productivité des développeurs en vous permettant de vous concentrer sur la création de fonctionnalités, tandis que VPC Lattice gère les défis liés au réseau, à la sécurité et à l'observabilité de manière uniforme sur toutes les plateformes de calcul.
- Meilleure posture de sécurité - VPC Lattice permet à vos développeurs d'authentifier et de sécuriser facilement les communications entre les applications et les plateformes informatiques, d'appliquer le chiffrement en transit et d'appliquer des contrôles d'accès granulaires grâce aux politiques d'authentification VPC Lattice. Cela vous permet d'adopter une posture de sécurité renforcée qui répond aux principales exigences réglementaires et de conformité du secteur.
- Évolutivité et résilience améliorées des applications - VPC Lattice vous permet de créer un réseau d'applications déployées avec des fonctionnalités telles que le routage, l'authentification, l'autorisation et la surveillance basés sur les chemins, les en-têtes et les méthodes. Ces avantages

sont fournis sans surcharge de ressources sur les charges de travail et peuvent prendre en charge des déploiements multi-clusters qui génèrent des millions de demandes par seconde sans ajouter de latence significative.

- Flexibilité de déploiement grâce à une infrastructure hétérogène - VPC Lattice fournit des fonctionnalités cohérentes sur tous les services de calcul tels qu'Amazon ECS, Fargate, EC2 Amazon, Amazon EKS et Lambda et permet à votre organisation de choisir l'infrastructure adaptée à chaque application.

Comment VPC Lattice fonctionne avec les autres services Amazon ECS

L'utilisation de VPC Lattice avec Amazon ECS peut modifier la façon dont vous utilisez les autres services Amazon ECS, tandis que les autres restent inchangés.

Application Load Balancers

Vous n'avez plus besoin de créer un Application Load Balancer spécifique à utiliser avec le type de groupe cible Application Load Balancer dans VPC Lattice, qui est ensuite lié au service Amazon ECS. Il vous suffit de configurer votre service Amazon ECS avec un groupe cible VPC Lattice à la place. Vous pouvez également choisir d'utiliser Application Load Balancer avec Amazon ECS en même temps.

Déploiements progressifs d'Amazon ECS

Seuls les déploiements progressifs d'Amazon ECS fonctionnent avec VPC Lattice, et Amazon ECS introduit des tâches dans les services et les supprime en toute sécurité pendant le déploiement. Le déploiement du code et les Blue/Green déploiements ne sont pas pris en charge.

Pour en savoir plus sur VPC Lattice, consultez le guide de l'utilisateur Amazon VPC Lattice.

Création d'un service utilisant VPC Lattice

Vous pouvez utiliser le AWS Management Console ou le AWS CLI pour créer un service avec VPC Lattice.

Prérequis

Avant de commencer ce didacticiel, vérifiez que vous respectez les conditions requises suivantes :

- La dernière version du AWS CLI est installée et configurée. Pour plus d’informations, consultez Installing the AWS Command Line Interface (Installation de).

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir de AWS CLI SDKs, et de l'API Amazon ECS via les deux IPv4 et. IPv6 Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

- Les étapes décrites dans Configurer l'utilisation d'Amazon ECS sont terminées.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique Amazon ECS_ FullAccess IAM.

Créez un service qui utilise VPC Lattice avec AWS Management Console

Procédez comme suit pour créer un service avec VPC Lattice à l'aide du. AWS Management Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la page de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster dans lequel créer le service.
4. Sous l'onglet Services choisissez Create (Créer).

   Si vous n'avez jamais créé de service auparavant, suivez les étapes décrites dans Création d'un service Amazon ECS à l'aide de la console, puis poursuivez ces étapes lorsque vous atteignez la section VPC Lattice.
5. Choisissez d'activer le réseau VPC en cochant le bouton.
6. Pour utiliser un rôle existant, pour le rôle d'infrastructure ECS pour Amazon ECS, choisissez-en un que vous avez déjà créé pour l'utiliser lors de la création du groupe cible VPC Lattice. Pour créer un nouveau rôle, créez un rôle d'infrastructure ECS.
7. Choisissez le VPC.

   Le VPC dépend du mode réseau que vous avez sélectionné lors de l'enregistrement de votre définition de tâche. Si vous utilisez le `network` mode `host` ou avec le type de EC2 lancement, choisissez votre VPC.

   Pour le `awsvpc` mode, le VPC est automatiquement sélectionné en fonction du VPC que vous avez choisi sous Mise en réseau et ne peut pas être modifié.

8. Sous Groupes cibles, sélectionnez le ou les groupes cibles. Vous devez choisir au moins un groupe cible et cinq au maximum. Choisissez Ajouter un groupe cible pour ajouter des groupes cibles supplémentaires. Choisissez le nom du port, le protocole et le port pour chaque groupe cible que vous avez choisi. Pour supprimer un groupe cible, choisissez Supprimer.

> **Note**
>
> - Si vous souhaitez ajouter des groupes cibles existants, vous devez utiliser le AWS CLI. Pour obtenir des instructions sur la façon d'ajouter des groupes cibles à l'aide de AWS CLI, voir register-targets dans la AWS Command Line Interface référence.
> - Bien qu'un service VPC Lattice puisse avoir plusieurs groupes cibles, chaque groupe cible ne peut être ajouté qu'à un seul service.

9. À ce stade, vous accédez à la console VPC Lattice pour poursuivre la configuration. C'est ici que vous incluez vos nouveaux groupes cibles dans l'action par défaut de l'écouteur ou dans les règles d'un service VPC Lattice existant.

   Pour plus d'informations, consultez la section Règles du récepteur pour votre service VPC Lattice.

> **Important**
>
> Vous devez autoriser le `vpc-lattice` préfixe de règle entrante à votre groupe de sécurité ou à vos tâches, car les contrôles de santé peuvent échouer.

Créez un service qui utilise VPC Lattice avec AWS CLI

Utilisez le AWS CLI pour créer un service avec VPC Lattice. Remplacez chaque *`user input placeholder`* par vos propres informations.

1. Créez un fichier de configuration du groupe cible. L'exemple suivant est nommé `tg-config.json`

```
{
    "ipAddressType": "IPV4",
    "port": 443,
    "protocol": "HTTPS",
```

```
    "protocolVersion": "HTTP1",
    "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
}
```

2. Utilisez la commande suivante pour créer un groupe cible VPC Lattice.

```
aws vpc-lattice create-target-group \
    --name my-lattice-target-group-ip \
    --type IP \
    --config file://tg-config.json
```

Exemple de sortie :

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",
    "config": {
        "healthCheck": {
            "enabled": true,
            "healthCheckIntervalSeconds": 30,
            "healthCheckTimeoutSeconds": 5,
            "healthyThresholdCount": 5,
            "matcher": {
                "httpCode": "200"
            },
            "path": "/",
            "protocol": "HTTPS",
            "protocolVersion": "HTTP1",
            "unhealthyThresholdCount": 2
        },
        "ipAddressType": "IPV4",
        "port": 443,
        "protocol": "HTTPS",
        "protocolVersion": "HTTP1",
        "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
    },
    "id": "tg-0eaa4b9ab4EXAMPLE",
    "name": "my-lattice-target-group-ip",
    "status": "CREATE_IN_PROGRESS",
    "type": "IP"
}
```

3. Le fichier JSON suivant *ecs-service-vpc-lattice.json* est un exemple utilisé pour associer un service Amazon ECS à un groupe cible VPC Lattice. `portName`Dans l'exemple ci-dessous, c'est le même que celui que vous avez défini dans le `name` champ de `portMappings` propriété de votre définition de tâche.

```
{
    "serviceName": "ecs-service-vpc-lattice",
    "taskDefinition": "ecs-task-def",
        "vpcLatticeConfigurations": [
        {
            "targetGroupArn": "arn:aws:vpc-lattice:us-
west-2:123456789012:targetgroup/tg-0eaa4b9ab4EXAMPLE",
            "portName": "testvpclattice",
            "roleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRoleVpcLattice"
        }
    ],
    "desiredCount": 5,
    "role": "ecsServiceRole"
}
```

Utilisez la commande suivante pour créer un service Amazon ECS et l'associer au groupe cible VPC Lattice à l'aide de l'exemple json ci-dessus.

```
aws ecs create-service \
    --cluster clusterName \
    --serviceName ecs-service-vpc-lattice \
    --cli-input-json file://ecs-service-vpc-lattice.json
```

> **Note**
>
> VPC Lattice n'est pas pris en charge sur Amazon ECS Anywhere.

Protégez vos tâches Amazon ECS pour éviter qu'elles ne soient interrompues par des événements évolutifs

Vous pouvez utiliser la protection évolutive des tâches d'Amazon ECS pour empêcher que vos tâches ne soient interrompues par des événements d'évolutivité liés au dimensionnement automatique des services ou à des déploiements.

Certaines applications nécessitent un mécanisme permettant de protéger les tâches critiques contre toute interruption due à des événements de mise à l'échelle horizontale en période de faible utilisation ou lors de déploiements de services. Par exemple :

- Vous disposez d'une application asynchrone de traitement des files d'attente, telle qu'une tâche de transcodage vidéo, dans laquelle certaines tâches doivent s'exécuter pendant des heures, même lorsque l'utilisation cumulée des services est faible.
- Vous disposez d'une application de jeu qui exécute des serveurs de jeux sous forme de tâches Amazon ECS qui doivent continuer à s'exécuter même si tous les utilisateurs sont déconnectés afin de réduire la latence de démarrage lors du redémarrage du serveur.
- Lorsque vous déployez une nouvelle version de code, vous avez besoin que des tâches continuent à s'exécuter, car leur retraitement serait coûteux.

Pour empêcher les tâches appartenant à votre service de se terminer lors d'un événement de mise à l'échelle horizontale, définissez l'attribut `ProtectionEnabled` sur `true`. Lorsque vous définissez `ProtectionEnabled` la valeur true, les tâches sont protégées pendant 2 heures par défaut. Vous pouvez ensuite personnaliser la période de protection à l'aide de l'`ExpiresInMinutes`attribut. Vous pouvez protéger vos tâches pendant au moins une minute et jusqu'à un maximum de 2 880 minutes (48 heures). Si vous utilisez le AWS CLI, vous pouvez spécifier l'`--protection-enabled`option.

Une fois qu'une tâche a terminé son travail requis, vous pouvez définir l'attribut `ProtectionEnabled` sur `false`, ce qui permet à la tâche d'être interrompue par des événements ultérieurs de mise à l'échelle horizontale. Si vous utilisez le AWS CLI, vous pouvez spécifier l'`--no-protection-enabled`option.

Mécanismes de protection évolutive des tâches

Vous pouvez définir et obtenir une protection évolutive des tâches à l'aide du point de terminaison de l'agent de conteneur Amazon ECS ou de l'API Amazon ECS.

- Point de terminaison de l'agent de conteneur Amazon ECS

Nous vous recommandons d'utiliser le point de terminaison de l'agent de conteneur Amazon ECS pour les tâches qui peuvent déterminer automatiquement le besoin de protection. Utilisez cette approche pour les charges de travail basées sur les files d'attente ou le traitement des tâches.

Lorsqu'un conteneur commence à traiter un travail, par exemple en consommant un message SQS, vous pouvez définir l'attribut `ProtectionEnabled` via le chemin `$ECS_AGENT_URI/task-protection/v1/state` du point de terminaison de protection évolutive des tâches depuis le conteneur. Amazon ECS n'interrompra pas cette tâche lors d'événements de mise à l'échelle horizontale. Une fois que votre tâche a terminé son travail, vous pouvez effacer l'`ProtectionEnabled`attribut en utilisant le même point de terminaison, afin que la tâche puisse être interrompue lors d'événements d'extension ultérieurs.

Pour plus d'informations sur le point de terminaison de l'agent de conteneur Amazon ECS, consultezPoint de terminaison de protection évolutif des tâches Amazon ECS.

- API Amazon ECS

  Vous pouvez utiliser l'API Amazon ECS pour définir et récupérer la protection évolutive des tâches si votre application possède un composant qui suit l'état des tâches actives. Utilisez `UpdateTaskProtection` pour marquer une ou plusieurs tâches comme protégées. `GetTaskProtection`À utiliser pour récupérer l'état de protection.

  Un exemple de cette approche serait si votre application héberge des sessions de serveur de jeu comme tâches Amazon ECS. Lorsqu'un utilisateur se connecte à une session sur le serveur (tâche), vous pouvez marquer la tâche comme protégée. Une fois que l'utilisateur se déconnecte, vous pouvez soit retirer la protection spécifique à cette tâche, soit retirer périodiquement la protection pour des tâches similaires qui ne comportent plus de sessions actives, en fonction de vos besoins en termes de serveurs inactifs.

  Pour plus d'informations, consultez UpdateTaskProtectionet consultez le GetTaskProtectionmanuel Amazon Elastic Container Service API Reference.

Vous pouvez combiner les deux approches. Par exemple, utilisez le point de terminaison de l'agent Amazon ECS pour définir la protection des tâches depuis un conteneur et utilisez l'API Amazon ECS pour retirer la protection des tâches depuis votre service de contrôleur externe.

Considérations

Prenez en compte les points suivants avant d'utiliser la protection évolutive des tâches :

- La protection intégrée des tâches n'est prise en charge que pour les tâches déployées à partir d'un service.
- Nous vous recommandons d'utiliser le point de terminaison de l'agent de conteneur Amazon ECS, car l'agent Amazon ECS possède des mécanismes de nouvelle tentative intégrés et une interface plus simple.
- Vous pouvez réinitialiser la période d'expiration de la protection contre la mise à l'échelle horizontale des tâches en appelant `UpdateTaskProtection` pour une tâche pour laquelle la protection est déjà activée.
- Déterminez le temps nécessaire à une tâche pour effectuer le travail requis et définissez la propriété `expiresInMinutes` en conséquence. Si vous fixez une période d'expiration de la protection plus longue que nécessaire, vous devrez supporter des coûts et serez confronté à des retards dans le déploiement de nouvelles tâches.
- La protection contre la mise à l'échelle horizontale des tâches est prise en charge sur l'agent de conteneur Amazon ECS version `1.65.0` ou version ultérieure.

  Vous pouvez ajouter la prise en charge de cette fonctionnalité sur les EC2 instances Amazon utilisant d'anciennes versions de l'agent de conteneur Amazon ECS en mettant à jour l'agent vers la dernière version. Pour de plus amples informations, veuillez consulter Mise à jour de l'agent de conteneur Amazon ECS.
- Considérations relatives au déploiement :
  - Si le service utilise une mise à jour propagée, de nouvelles tâches seront créées, mais les tâches exécutant une ancienne version ne seront pas terminées avant la désactivation ou l'expiration de `protectionEnabled`. Vous pouvez ajuster le paramètre `maximumPercentage` dans la configuration du déploiement sur une valeur qui permet de créer de nouvelles tâches lorsque les anciennes tâches sont protégées.
  - Si une blue/green mise à jour est appliquée, le déploiement bleu avec des tâches protégées ne sera pas supprimé si des tâches l'ont été`protectionEnabled`. Le trafic sera redirigé vers les nouvelles tâches qui apparaîtront et les anciennes ne seront supprimées que lorsqu'elles `protectionEnabled` seront effacées ou expireront. En fonction du délai d'expiration des CodeDeploy CloudFormation mises à jour, le déploiement peut expirer et les anciennes tâches Blue peuvent toujours être présentes.
  - Si vous l'utilisez CloudFormation, le délai d'expiration de la pile de mises à jour est de 3 heures. Par conséquent, si vous définissez la protection des tâches pour une durée supérieure à 3 heures, votre CloudFormation déploiement peut entraîner un échec et une annulation.

Pendant le temps que vos anciennes tâches sont protégées, la CloudFormation pile apparaît`UPDATE_IN_PROGRESS`. Si la protection contre la mise à l'échelle horizontale des tâches est supprimée ou expire dans un délai de trois heures, votre déploiement réussira et passera au statut `UPDATE_COMPLETE`. Si le déploiement est bloqué dans l'état `UPDATE_IN_PROGRESS` pendant plus de 3 heures, il échouera et affichera un état `UPDATE_FAILED`, puis reviendra au dernier ensemble de tâches.

- Amazon ECS envoie des événements de service lorsque des tâches protégées empêchent un déploiement (propagé ou bleu/vert) d'atteindre l'état stable, afin que vous puissiez prendre des mesures correctives. Lorsque vous essayez de mettre à jour l'état de protection d'une tâche et que vous recevez un message d'erreur `DEPLOYMENT_BLOCKED`, cela signifie que le service possède un nombre de tâches protégées supérieur au nombre de tâches souhaité pour le service. Pour résoudre cette erreur, effectuez l'une des opérations suivantes :
  - Attendez que la protection des tâches en cours expire. Définissez ensuite la protection des tâches.
  - Déterminez quelles tâches peuvent être interrompues. Utilisez ensuite `UpdateTaskProtection` avec l'option `protectionEnabled` définie sur `false` pour ces tâches.
  - Augmentez le nombre de tâches souhaité pour le service à un nombre supérieur au nombre de tâches protégées.

Autorisations IAM requises pour la protection évolutive des tâches

La tâche doit avoir le rôle de tâche Amazon ECS avec les autorisations suivantes :

- `ecs:GetTaskProtection` : permet à l'agent de conteneur Amazon ECS d'appeler `GetTaskProtection`.
- `ecs:UpdateTaskProtection` : permet à l'agent de conteneur Amazon ECS d'appeler `UpdateTaskProtection`.

Point de terminaison de protection évolutif des tâches Amazon ECS

L'agent de conteneur Amazon ECS injecte automatiquement la variable d'environnement `ECS_AGENT_URI` dans les conteneurs des tâches Amazon ECS afin de fournir une méthode permettant d'interagir avec le point de terminaison de l'API de l'agent de conteneur.

Nous vous recommandons d'utiliser le point de terminaison de l'agent de conteneur Amazon ECS pour les tâches qui peuvent déterminer automatiquement le besoin de protection.

Lorsqu'un conteneur commence à traiter des tâches, vous pouvez définir l'`protectionEnabled`attribut à l'aide du chemin du point de terminaison de protection évolutif des tâches `$ECS_AGENT_URI/task-protection/v1/state` depuis le conteneur.

Utilisez une requête PUT envoyée à cet URI depuis un conteneur pour définir la protection intégrée des tâches. Une requête GET envoyée à cet URI renvoie l'état de protection actuel d'une tâche.

Paramètres de demande de protection évolutifs des tâches

Vous pouvez définir la protection évolutive des tâches à l'aide du point de terminaison `${ECS_AGENT_URI}/task-protection/v1/state` avec les paramètres de demande suivants.

`ProtectionEnabled`

Spécifiez `true` pour marquer une tâche à protéger. Spécifiez `false` si vous souhaitez supprimer la protection et rendre la tâche éligible à l'arrêt.

Type : booléen

Obligatoire : oui

`ExpiresInMinutes`

Le nombre de minutes pendant lesquelles la tâche est protégée. Vous pouvez spécifier un minimum de 1 minute et aller jusqu'à 2 880 minutes (48 heures). Pendant cette période, votre tâche ne sera pas interrompue par des événements de mise à l'échelle horizontale provenant de l'autoscaling du service ou de déploiements. Une fois cette période écoulée, le paramètre `protectionEnabled` est défini sur `false`.

Si vous ne spécifiez pas l'heure, la tâche est automatiquement protégée pendant 120 minutes (2 heures).

Type : entier

Obligatoire : non

Les exemples suivants montrent comment définir la protection des tâches avec des durées différentes.

Exemple de protection d'une tâche avec la période par défaut

Cet exemple montre comment protéger une tâche dont la durée par défaut est de 2 heures.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true}'
```

Exemple de protection d'une tâche pendant 60 minutes

Cet exemple montre comment protéger une tâche pendant 60 minutes à l'aide du paramètre `expiresInMinutes`.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true,"ExpiresInMinutes":60}'
```

Exemple de protection d'une tâche pendant 24 heures

Cet exemple montre comment protéger une tâche pendant 24 heures à l'aide du paramètre `expiresInMinutes`.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true,"ExpiresInMinutes":1440}'
```

Exemples de conteneurs Windows

Pour les conteneurs Windows, vous pouvez utiliser l'`Invoke-RestMethod`applet PowerShell de commande au lieu de curl. Les exemples suivants montrent les PowerShell équivalents des commandes curl précédentes.

Exemple de protection d'une tâche de conteneur Windows avec la période par défaut

Cet exemple montre comment protéger une tâche dont la durée par défaut est de 2 heures en utilisant PowerShell.

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Put -Body
 '{"ProtectionEnabled":true}' -ContentType 'application/json'
```

Exemple de protection d'une tâche de conteneur Windows pendant 60 minutes

Cet exemple montre comment protéger une tâche pendant 60 minutes à l'aide du `expiresInMinutes` paramètre with PowerShell.

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Put -Body
 '{"ProtectionEnabled":true,"ExpiresInMinutes":60}' -ContentType 'application/json'
```

Exemple de protection d'une tâche de conteneur Windows pendant 24 heures

Cet exemple montre comment protéger une tâche pendant 24 heures à l'aide du `expiresInMinutes` paramètre with PowerShell.

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Put -Body
 '{"ProtectionEnabled":true,"ExpiresInMinutes":1440}' -ContentType 'application/json'
```

La requête PUT renvoie la réponse suivante.

```
{
  "protection": {
    "ExpirationDate": "2023-12-20T21:57:44.837Z",
    "ProtectionEnabled": true,
    "TaskArn": "arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0"
  }
}
```

Paramètres de réponse de protection évolutifs des tâches

Les informations suivantes sont renvoyées à partir de la réponse JSON du point de terminaison de protection contre la mise à l'échelle horizontale des tâches `${ECS_AGENT_URI}/task-protection/v1/state`.

`ExpirationDate`

Période pendant laquelle la protection de la tâche expirera. Si la tâche n'est pas protégée, cette valeur est nulle.

`ProtectionEnabled`

État de protection de la tâche. Si la protection évolutive est activée pour une tâche, la valeur est `true`. Sinon, la valeur est `false`.

`TaskArn`

Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

L'exemple suivant affiche les détails renvoyés pour une tâche protégée.

```
curl --request GET ${ECS_AGENT_URI}/task-protection/v1/state
```

Pour les conteneurs Windows, utilisez la PowerShell commande suivante pour obtenir l'état de protection :

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Get
```

```
{
    "protection":{
        "ExpirationDate":"2023-12-20T21:57:44Z",
        "ProtectionEnabled":true,
        "TaskArn":"arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0"
    }
}
```

Les informations suivantes sont renvoyées en cas d'échec.

`Arn`

Amazon Resource Name (ARN) complet de la tâche.

`Detail`

Détails relatifs à l'échec.

`Reason`

Raison de l'échec.

L'exemple suivant affiche les détails renvoyés pour une tâche qui n'est pas protégée.

```
{
    "failure":{
        "Arn":"arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0",
        "Detail":null,
        "Reason":"TASK_NOT_VALID"
    }
}
```

Les informations suivantes sont renvoyées en cas d'exception.

`requestID`

L'ID de AWS demande pour l'appel d'API Amazon ECS qui entraîne une exception.

`Arn`

Amazon Resource Name (ARN) complet de la tâche ou du service.

`Code`

Code de l'erreur.

`Message`

Message d'erreur.

> **Note**
>
> Si une erreur `RequestError` ou `RequestTimeout` apparaît, il s'agit probablement d'un problème de mise en réseau. Essayez d'utiliser les points de terminaison d'un VPC pour Amazon ECS.

L'exemple suivant affiche les détails renvoyés en cas d'erreur.

```
{
    "requestID":"12345-abc-6789-0123-abc",
    "error":{
        "Arn":"arn:aws:ecs:us-west-2:555555555555:task/my-cluster-
name/1234567890abcdef0",
        "Code":"AccessDeniedException",
        "Message":"User: arn:aws:sts::444455556666:assumed-role/my-ecs-task-
role/1234567890abcdef0 is not authorized to perform: ecs:GetTaskProtection on resource:
 arn:aws:ecs:us-west-2:555555555555:task/test/1234567890abcdef0 because no identity-
based policy allows the ecs:GetTaskProtection action"
    }
}
```

L'erreur suivante s'affiche si l'agent Amazon ECS ne parvient pas à obtenir de réponse du point de terminaison Amazon ECS pour des raisons telles que des problèmes de réseau ou si le plan de contrôle Amazon ECS est en panne.

```
{
  "error": {
```

```
    "Arn": "arn:aws:ecs:us-west-2:555555555555:task/my-cluster-name/1234567890abcdef0",
    "Code": "RequestCanceled",
    "Message": "Timed out calling Amazon ECS Task Protection API"
  }
}
```

L'erreur suivante apparaît lorsque l'agent Amazon ECS reçoit une exception de limitation de la part d'Amazon ECS.

```
{
  "requestID": "12345-abc-6789-0123-abc",
  "error": {
    "Arn": "arn:aws:ecs:us-west-2:555555555555:task/my-cluster-name/1234567890abcdef0",
    "Code": "ThrottlingException",
    "Message": "Rate exceeded"
  }
}
```

Utilisez l'injection de défauts avec vos charges de travail Amazon ECS et Fargate

Les clients peuvent utiliser l'injection d'erreurs avec Amazon ECS sur Amazon EC2 et Fargate pour tester la manière dont leur application répond à certains scénarios de défaillance. Ces tests fournissent des informations que vous pouvez utiliser pour optimiser les performances et la résilience de votre application.

Lorsque l'injection d'erreurs est activée, l'agent de conteneur Amazon ECS permet aux tâches d'accéder aux nouveaux points de terminaison d'injection de pannes. Vous devez vous inscrire pour utiliser l'injection de défauts en définissant la valeur du paramètre de définition de la `enableFaultInjection` tâche sur`true`. La valeur par défaut est `false`.

```
{
    ...
   "enableFaultInjection": true
}
```

> **Note**
>
> L'injection de défauts ne fonctionne qu'avec les tâches utilisant le mode `awsvpc` ou le mode `host` réseau.

L'injection de défauts n'est pas disponible sous Windows.

Pour plus d'informations sur la façon d'activer l'injection de défauts dans le AWS Management Console, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Vous devez activer la fonctionnalité pour le tester dans le AWS Fault Injection Service. Pour plus d'informations, voir Utiliser les actions AWS FIS aws:ecs:task.

> **Note**
>
> Si vous n'utilisez pas le nouvel Amazon ECS optimisé AMIs ou si vous ne disposez pas d'une AMI personnalisée, installez les dépendances suivantes :
>
> - `tc`
> - `sch_netem`module noyau

Points de terminaison d'injection de pannes Amazon ECS

L'agent de conteneur Amazon ECS injecte automatiquement la variable d'environnement `ECS_AGENT_URI` dans les conteneurs des tâches Amazon ECS afin de fournir une méthode permettant d'interagir avec le point de terminaison de l'API de l'agent de conteneur. Chaque point de terminaison inclut un point de terminaison `/start/stop`, et un point de `/status` terminaison. Les points de terminaison n'acceptent que les demandes provenant de tâches qui ont activé l'injection d'erreurs, et chaque point de terminaison est limité à une demande toutes les 5 secondes par conteneur. Le dépassement de cette limite entraîne une erreur.

> **Note**
>
> L'agent Amazon ECS `version 1.88.0+` est requis pour utiliser les points de terminaison d'injection d'erreurs.

Les trois points de terminaison à utiliser avec l'injection de défauts sont les suivants :

- Point de terminaison du port Blackhole du réseau
- Point de terminaison de perte de paquets réseau

- Point de latence du réseau

Une demande réussie génère un code de réponse `200` contenant un message indiquant `running` quand vous appelez le `/start` point de terminaison, `stopped` pour le point de `/stop` terminaison `running` et/ou `not-running` pour le `/status` point de terminaison.

```
{
    "Status": <string>
}
```

Une demande infructueuse renvoie l'un des codes d'erreur suivants :

- `400`- Mauvaise demande
- `409`- La demande d'injection de défauts est en conflit avec une autre erreur en cours
- `429`- La demande a été limitée
- `500`- Une erreur inattendue s'est produite sur le serveur

```
{
 "Error":  <string message>
}
```

> **Note**
>
> Un défaut de latence réseau ou un défaut de perte de paquets réseau peuvent être injectés à la fois. Si vous essayez d'en injecter plusieurs, la demande est rejetée.

Point de terminaison du port Blackhole du réseau

Le `{ECS_AGENT_URI}/fault/v1/network-blackhole-port` point de terminaison supprime le trafic entrant ou sortant pour un port et un protocole spécifiques dans l'espace de noms réseau d'une tâche et est compatible avec deux modes :

- awsvpc - les modifications sont appliquées à l'espace de noms du réseau de tâches
- host - les modifications sont appliquées à l'instance de conteneur d'espace de noms réseau par défaut

{ECS_AGENT_URI}/fault/v1/network-blackhole-port/start

Ce point de terminaison lance les injections de défauts du port Blackhole du réseau et possède les paramètres suivants :

Port

Port spécifié à utiliser pour l'injection des défauts du port Blackhole.

Type : entier

Obligatoire : oui

Protocole

Protocole à utiliser pour l'injection de défauts dans le port Blackhole.

Type : String

Valeurs valides : `tcp | udp`

Obligatoire : oui

TrafficType

Type de trafic utilisé par l'injection de défauts.

Type : String

Valeurs valides : `ingress | egress`

Obligatoire : oui

SourcesToFilter

Tableau JSON d' IPv4 adresses ou de blocs CIDR protégés contre le défaut.

Type : tableau de chaînes

Obligatoire : non

Voici un exemple de demande d'utilisation du `start` point de terminaison (remplacez les *`red`* valeurs par les vôtres) :

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-blackhole-port/start

Http method:POST

Request payload:
{
    "Port": 1234,
    "Protocol": "tcp|udp",
    "TrafficType": "ingress|egress"
    "SourcesToFilter": ["${IP1}", "${IP2}", ...],
}
```

{ECS_AGENT_URI}/fault/v1/network-blackhole-port/stop

Ce point de terminaison arrête le défaut spécifié dans la demande. Ce point de terminaison possède les paramètres suivants :

Port

Port concerné par le défaut qui doit être arrêté.

Type : entier

Obligatoire : oui

Protocole

Protocole à utiliser pour arrêter le défaut.

Type : String

Valeurs valides : `tcp | udp`

Obligatoire : oui

TrafficType

Type de trafic utilisé par l'injection de défauts.

Type : String

Valeurs valides : `ingress | egress`

Obligatoire : oui

Voici un exemple de demande d'utilisation du `stop` point de terminaison (remplacez les *`red`* valeurs par les vôtres) :

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-blackhole-port/stop

Http method: POST

Request payload:
{
    "Port": 1234,
    "Protocol": "tcp|udp",
    "TrafficType": "ingress|egress",
}
```

{ECS_AGENT_URI}/fault/v1/network-blackhole-port/status

Ce point de terminaison est utilisé pour vérifier l'état de l'injection de défauts. Ce point de terminaison possède les paramètres suivants :

Port

Le port concerné pour vérifier l'état du défaut.

Type : entier

Obligatoire : oui

Protocole

Protocole à utiliser lors de la vérification de l'état du défaut.

Type : String

Valeurs valides : `tcp | udp`

Obligatoire : oui

TrafficType

Type de trafic utilisé par l'injection de défauts.

Type : String

Valeurs valides : `ingress | egress`

Obligatoire : oui

Voici un exemple de demande d'utilisation du `status` point de terminaison (remplacez les *`red`* valeurs par les vôtres) :

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-blackhole-port/status

Http method: POST

Request payload:
{
   "Port": 1234,
   "Protocol": "tcp|udp",
   "TrafficType": "ingress|egress",
}
```

Point de latence du réseau

Le `{ECS_AGENT_URI}/fault/v1/network-latency` point de terminaison ajoute du retard et de la gigue à l'interface réseau de la tâche pour le trafic vers une source spécifique. Le terminal est compatible avec deux modes :

- awsvpc - les modifications sont appliquées à l'interface réseau des tâches
- hôte - les modifications sont appliquées à l'interface réseau par défaut

{ECS_AGENT_URI}/fault/v1/network-latency/start

Ce `/start` point de terminaison commence l'injection des défauts de latence du réseau et possède les paramètres suivants :

DelayMilliseconds

Le nombre de millisecondes de délai à ajouter à l'interface réseau à utiliser pour l'injection de défauts.

Type : entier

Obligatoire : oui

JitterMilliseconds

Le nombre de millisecondes de gigue à ajouter à l'interface réseau à utiliser pour l'injection de défauts.

Type : entier

Obligatoire : oui

Sources

Tableau JSON d' IPv4 adresses ou de blocs CIDR destinés à être utilisés pour l'injection d'erreurs.

Type : tableau de chaînes

Obligatoire : oui

SourcesToFilter

Tableau JSON d' IPv4 adresses ou de blocs CIDR protégés contre le défaut. `SourcesToFiltera` la priorité sur`Sources`.

Type : tableau de chaînes

Obligatoire : non

Voici un exemple de demande d'utilisation du `/start` point de terminaison (remplacez les *`red`* valeurs par les vôtres) :

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-latency/start

Http method: POST

Request payload:
{
    "DelayMilliseconds": 123,
    "JitterMilliseconds": 123,
    "Sources": ["${IP1}", "${IP2}", ...],
    "SourcesToFilter": ["${IP1}", "${IP2}", ...],
}
```

{ECS_AGENT_URI}/fault/v1/network-latency/stop and /status

Le `{ECS_AGENT_URI}/fault/v1/network-latency/stop` point de terminaison arrête le défaut et `{ECS_AGENT_URI}/fault/v1/network-latency/status` vérifie l'état du défaut.

Voici deux exemples de demandes d'utilisation des `/status` points de terminaison `/stop` et. Les deux utilisent la `POST` `HTTP` méthode.

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-latency/stop
```

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-latency/status
```

Point de terminaison de perte de paquets réseau

Le `{ECS_AGENT_URI}/fault/v1/network-packet-loss` point de terminaison ajoute la perte de paquets à l'interface réseau donnée. Ce terminal est compatible avec deux modes :

- awsvpc - les modifications sont appliquées à l'interface réseau des tâches
- hôte - les modifications sont appliquées à l'interface réseau par défaut

{ECS_AGENT_URI}/fault/v1/network-packet-loss/start

Ce `/start` point de terminaison lance l'injection d'erreur de perte de paquets réseau et possède les paramètres suivants :

LossPercent

Le pourcentage de perte de paquets

Type : entier

Obligatoire : oui

Sources

Un tableau JSON d' IPv4 adresses ou de blocs CIDR à utiliser pour les tests d'injection d'erreurs.

Type : tableau de chaînes

Obligatoire : oui

SourcesToFilter

Tableau JSON d' IPv4 adresses ou de blocs CIDR protégés contre le défaut. `SourcesToFilter`a la priorité sur`Sources`.

Type : tableau de chaînes

Obligatoire : non

Voici un exemple de demande d'utilisation du `start` point de terminaison (remplacez les *`red`* valeurs par les vôtres) :

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-packet-loss/start

Http method: POST

{
    "LossPercent": 6,
    "Sources": ["${IP1}", "${IP2}", ...],
    "SourcesToFilter": ["${IP1}", "${IP2}", ...],
}
```

{ECS_AGENT_URI}/fault/v1/network-packet-loss/stop and /status

Le `{ECS_AGENT_URI}/fault/v1/network-packet-loss/stop` point de terminaison arrête le défaut et `{ECS_AGENT_URI}/fault/v1/network-packet-loss/status` vérifie l'état du défaut. Un seul de chaque type de défaut est pris en charge à la fois.

Voici deux exemples de demandes d'utilisation des `/status` points de terminaison `/stop` et. Les deux utilisent la `POST HTTP` méthode.

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-packet-loss/stop
```

```
Endpoint: ${{ECS_AGENT_URI}/fault/v1/network-packet-loss/status
```

Mettre à jour les paramètres du service Amazon ECS

Après avoir créé un service, il peut arriver que vous deviez mettre à jour les paramètres du service, par exemple le nombre de tâches.

Lorsque le planificateur de services lance de nouvelles tâches, il détermine le placement des tâches dans votre cluster selon la logique suivante.

- Déterminez les instances de conteneur de votre cluster qui peuvent prendre en charge la définition des tâches de votre service. Par exemple, ils possèdent le processeur, la mémoire, les ports et les attributs d'instance de conteneur requis.

- Par défaut, le planificateur de services tente d'équilibrer les tâches entre les zones de disponibilité de cette manière, même si vous pouvez choisir une autre stratégie de placement.
  - Triez les instances de conteneur valides selon le plus petit nombre de tâches en cours d'exécution pour ce service dans la même zone de disponibilité que l'instance. Par exemple, si la zone A comporte une tâche de service en cours d'exécution tandis que les zones B et C n'en ont pas, les instances de conteneur valides des zones B ou C sont considérées comme optimales pour le placement.
  - Placer la nouvelle tâche de service sur une instance de conteneur valide dans une zone de disponibilité optimale (sur la base des étapes précédentes), en préférant les instances de conteneur avec le plus petit nombre de tâches en cours d'exécution pour ce service.

Lorsque le planificateur de services arrête d'exécuter des tâches, il tente de maintenir l'équilibre entre les zones de disponibilité de votre cluster selon la logique suivante :

- Triez les instances de conteneur selon le plus grand nombre de tâches en cours d'exécution pour ce service dans la même zone de disponibilité que l'instance. Par exemple, si la zone A comporte une tâche de service en cours d'exécution tandis que les zones B et C en ont chacune deux, les instances de conteneur des zones B ou C sont considérées comme optimales pour la mise hors service.
- Arrêter la nouvelle tâche de service sur une instance de conteneur valide dans une zone de disponibilité optimale (sur la base des étapes précédentes), en préférant les instances de conteneur avec le plus grand nombre de tâches en cours d'exécution pour ce service.

Utilisez la liste pour déterminer si vous pouvez modifier le paramètre de service.

Rééquilibrage des zones de disponibilité

Indique s'il faut utiliser le rééquilibrage des zones de disponibilité pour le service.

Vous pouvez modifier ce paramètre pour les déploiements progressifs.

Stratégie des fournisseurs de capacités

Détails d'une stratégie de fournisseur de capacité. Vous pouvez définir un fournisseur de capacité lorsque vous créez un cluster, exécutez une tâche ou mettez à jour un service.

Lorsque vous utilisez Fargate, les fournisseurs de capacité sont ou. `FARGATE` `FARGATE_SPOT`

Lorsque vous utilisez Amazon EC2, les fournisseurs de capacité sont des groupes Auto Scaling.

Vous pouvez changer de fournisseur de capacité pour les déploiements progressifs et les déploiements bleu/vert.

La liste suivante fournit les transitions valides :

- Mettez à jour le type de lancement Fargate vers un fournisseur de capacité de groupe Auto Scaling.
- Mettez à jour le type de EC2 lancement Amazon vers un fournisseur de capacité Fargate.
- Transformez le fournisseur de capacité Fargate en fournisseur de capacité du groupe Auto Scaling.
- Transformez le fournisseur EC2 de capacité Amazon en fournisseur de capacité Fargate.
- Rétablissez le type de lancement du groupe Auto Scaling ou du fournisseur de capacité Fargate. Lorsque vous utilisez la CLI ou l'API, vous transmettez une liste vide dans le `capacityProviderStrategy` paramètre.

Cluster

Vous ne pouvez pas modifier le nom du cluster.

Configuration de déploiement

La configuration de déploiement inclut les CloudWatch alarmes, le disjoncteur utilisé pour détecter les défaillances et la configuration requise.

La disjoncteur de circuit de déploiement détermine si un déploiement de service échoue dans le cas où le service ne parvient pas à atteindre un état stable. Si vous utilisez le disjoncteur de déploiement, le déploiement d'un service passera à l'état d'échec et cessera de lancer de nouvelles tâches. Si vous utilisez l'option de restauration, lorsqu'un déploiement de service échoue, le service est annulé jusqu'au dernier déploiement terminé avec succès.

Lorsque vous mettez à jour un service qui utilise un disjoncteur Amazon ECS, Amazon ECS crée un déploiement et une révision du service. Ces ressources vous permettent de consulter des informations détaillées sur l'historique des services. Pour de plus amples informations, veuillez consulter Afficher l'historique des services à l'aide des déploiements de services Amazon ECS.

Le planificateur de service utilise les paramètres de pourcentage minimum d'instances saines et de pourcentage maximal (dans la configuration de déploiement pour le service) afin de déterminer la stratégie de déploiement.

Si un service utilise le type de déploiement rolling update (`ECS`), le pourcentage d'intégrité minimal représente une limite inférieure du nombre de tâches d'un service qui doivent rester `RUNNING`

inchangées pendant un déploiement, en pourcentage du nombre de tâches souhaité (arrondi au nombre entier le plus proche). Le paramètre s'applique également lorsque toutes les instances de conteneur sont dans l'`DRAINING`état où le service contient des tâches utilisant le type de EC2 lancement. Utilisez ce paramètre pour procéder au déploiement sans avoir recours à une capacité de cluster supplémentaire. Par exemple, si votre service comporte un nombre souhaité de tâches égal à quatre et un pourcentage minimum d'instances saines de 50 pour cent, le planificateur pourra arrêter deux tâches existantes afin de libérer de la capacité de cluster avant de lancer deux nouvelles tâches. Le service considère que les tâches sont saines pour les services qui n'utilisent pas d'équilibreur de charge s'ils sont dans `RUNNING` cet état. Le service considère que les tâches sont saines pour les services qui utilisent un équilibreur de charge s'ils sont dans `RUNNING` cet état et qu'ils sont signalés comme sains par l'équilibreur de charge. La valeur par défaut pour le pourcentage minimum d'instances saines est 100 pour cent.

Si un service utilise le type de déploiement rolling update (`ECS`), le paramètre de pourcentage maximal représente la limite supérieure du nombre de tâches autorisées dans un service dans l'`STOPPING`état `PENDINGRUNNING`, ou pendant un déploiement, sous forme de pourcentage du nombre de tâches souhaité (arrondi à l'entier inférieur le plus proche). Le paramètre s'applique également lorsque toutes les instances de conteneur sont dans l'`DRAINING`état où le service contient des tâches utilisant le type de EC2 lancement. Utilisez ce paramètre pour définir la taille des lots de déploiement. Par exemple, si votre service a un nombre souhaité de quatre tâches et une valeur de pourcentage maximum de 200 pour cent, le planificateur peut lancer quatre nouvelles tâches avant d'arrêter les quatre tâches plus anciennes. Cela s'applique à condition que les ressources de cluster nécessaires à cette opération soient disponibles. La valeur par défaut pour le pourcentage maximal est 200 pour cent.

Lorsque le planificateur de service remplace une tâche pendant une mise à jour, si un équilibreur de charge est utilisé par le service, ce dernier supprime tout d'abord la tâche de l'équilibreur de charge (le cas échéant) et attend que les connexions s'épuisent. Ensuite, l'équivalent de docker stop est émis pour les conteneurs en cours d'exécution dans la tâche. Cela se traduit par un signal `SIGTERM` et un délai de 30 secondes, après quoi `SIGKILL` est envoyé et les conteneurs sont arrêtés de force. Si le conteneur gère le signal `SIGTERM` normalement et s'arrête dans les 30 secondes suivant sa réception, aucun signal `SIGKILL` n'est envoyé. Le planificateur de service lance et arrête les tâches selon les modalités définies dans vos paramètres de pourcentage minimum d'instances saines et de pourcentage maximal.

Le planificateur de services remplace également les tâches jugées défectueuses après l'échec d'une surveillance de l'état du conteneur ou du groupe cible de l'équilibreur de charge. Ce remplacement dépend des paramètres de définition du service `maximumPercent` et

`desiredCount`. Si une tâche est marquée comme défectueuse, le planificateur de services lance d'abord une tâche de remplacement. Ensuite, ce qui suit se produit.

- Si l'état de santé de la tâche de remplacement est égal à`HEALTHY`, le planificateur de services arrête la tâche défectueuse
- Si l'état de santé de la tâche de remplacement est `UNHEALTHY`, le planificateur arrête soit la tâche de remplacement défectueuse, soit la tâche défectueuse existante pour que le nombre total de tâches soit égal à `desiredCount`.

Si le paramètre `maximumPercent` empêche le planificateur de démarrer d'abord une tâche de remplacement, le planificateur arrête une par une les tâches défectueuses au hasard pour libérer de la capacité, puis lance une tâche de remplacement. Le processus de démarrage et d'arrêt se poursuit jusqu'à ce que toutes les tâches défectueuses soient remplacées par des tâches saines. Une fois que toutes les tâches défectueuses ont été remplacées et que seules les tâches saines sont en cours d'exécution, si le nombre total de tâches dépasse le `desiredCount`, les tâches saines sont arrêtées au hasard jusqu'à ce que le nombre total de tâches soit égal à `desiredCount`. Pour plus d'informations sur `maximumPercent` et `desiredCount`, veuillez consulter Paramètres de définition de service (langue française non garantie).

Contrôleur de déploiement

Le contrôleur de déploiement à utiliser pour le service. Il existe trois types de contrôleurs de déploiement disponibles :

- `ECS`
- `EXTERNAL`
- `CODE_DEPLOY`

Lorsque vous mettez à jour un service, vous pouvez mettre à jour le contrôleur de déploiement qu'il utilise. La liste suivante fournit les transitions valides :

- Passez des déploiements CodeDeploy bleu/vert (`CODE_DEPLOY`) au déploiement ou aux blue/green déploiements ECS (). `ECS`
- Passez des déploiements CodeDeploy bleu/vert (`CODE_DEPLOY`) aux déploiements externes (). `EXTERNAL`
- Mise à jour depuis le déploiement ou les blue/green déploiements d'ECS (`ECS`) vers des déploiements externes (`EXTERNAL`).
- Mise à jour depuis les déploiements externes (`EXTERNAL`) vers le blue/green déploiement ou les déploiements ECS ()`ECS`.

Tenez compte des points suivants lorsque vous mettez à jour le contrôleur de déploiement d'un service :

- Vous ne pouvez pas mettre à jour le contrôleur de déploiement d'un service depuis le contrôleur de `ECS` déploiement vers un autre contrôleur s'il utilise VPC Lattice ou Amazon ECS Service Connect.
- Vous ne pouvez pas mettre à jour le contrôleur de déploiement d'un service pendant le déploiement d'un service en cours.
- Vous ne pouvez pas mettre à jour le contrôleur de déploiement d'un service `CODE_DEPLOY` s'il n'existe aucun équilibreur de charge sur le service.
- Vous ne pouvez pas mettre à jour le contrôleur de déploiement d'un service `ECS` vers un autre contrôleur s'il `deploymentConfiguration` inclut des alarmes, un disjoncteur de déploiement ou une stratégie de `BLUE_GREEN` déploiement. Pour de plus amples informations, veuillez consulter Contrôles et stratégies de déploiement des services Amazon ECS.
- La valeur que vous spécifiez `versionConsistency` dans la définition du conteneur ne sera pas utilisée par Amazon ECS si vous mettez à jour le contrôleur de déploiement du service `ECS` vers l'un des autres contrôleurs.
- Si vous mettez à jour le contrôleur de déploiement d'un service `ECS` vers l'un des autres contrôleurs, les réponses de l'`DescribeService`API `UpdateService` et seront toujours renvoyées `deployments` au lieu de`taskSets`. Pour plus d'informations sur `UpdateService` et`CreateService`, consultez UpdateServiceet CreateServicedans le manuel Amazon ECS API Reference.
- Si un service utilise une stratégie de déploiement de mise à jour continue, la mise à jour du contrôleur de déploiement `ECS` vers l'un des autres contrôleurs modifiera la façon dont la `maximumPercent` valeur du `deploymentConfiguration` est utilisée. Au lieu d'être simplement utilisé pour limiter le nombre total de tâches dans le cadre d'un déploiement de mises à jour continues, `maximumPercent` il est utilisé pour remplacer les tâches non fonctionnelles. Pour plus d'informations sur la façon dont le planificateur remplace les tâches non fonctionnelles, consultez. Services Amazon ECS
- Si vous mettez à jour le contrôleur de déploiement d'un service `ECS` vers l'un des autres contrôleurs de déploiement, ceux `advancedConfiguration` que vous spécifiez dans la configuration de votre équilibreur de charge seront ignorés. Pour plus d'informations, consultez LoadBalanceret consultez AdvancedConfigurationla référence de l'API Amazon ECS.

Lorsque vous mettez à jour le contrôleur de déploiement pour un service utilisant CloudFormation, tenez compte des points suivants en fonction du type de migration que vous effectuez.

- Si vous avez un CloudFormation modèle qui contient les informations du contrôleur de `EXTERNAL` déploiement ainsi que les `PrimaryTaskSet` ressources `TaskSet` et que vous supprimez les ressources de l'ensemble de tâches du modèle lors de la mise à jour de `EXTERNAL` vers`ECS`, les appels d'`DeleteTaskSet`API `DescribeTaskSet` et renverront une erreur 400 une fois le contrôleur de déploiement mis à jour vers`ECS`. Cela entraîne un échec de CloudFormation suppression des ressources de l'ensemble de tâches, même si la CloudFormation pile passe au `UPDATE_COMPLETE` statut. Pour plus d'informations, voir Ressource supprimée de la pile mais non supprimée dans le guide de AWS CloudFormation l'utilisateur. Pour résoudre ce problème, supprimez les ensembles de tâches directement à l'aide de l'`DeleteTaskSet`API Amazon ECS. Pour plus d'informations sur la suppression d'un ensemble de tâches, consultez DeleteTaskSetle manuel Amazon Elastic Container Service API Reference.
- Si vous effectuez une migration de `CODE_DEPLOY` vers `ECS` avec une nouvelle définition de tâche et que vous CloudFormation effectuez une opération de restauration, la `UpdateService` demande Amazon ECS échoue avec l'erreur suivante :

  ```
  Resource handler returned message: "Invalid request provided: Unable to update
  task definition on services with a CODE_DEPLOY deployment controller. Use AWS
  CodeDeploy to trigger a new deployment. (Service: Ecs, Status Code: 400,
  Request ID: 0abda1e2-f7b3-4e96-b6e9-c8bc585181ac) (SDK Attempt Count: 1)"
  (RequestToken: ba8767eb-c99e-efed-6ec8-25011d9473f0, HandlerErrorCode:
   InvalidRequest)
  ```

- Après une migration réussie depuis `ECS` `EXTERNAL` Deployment Controller, vous devez supprimer manuellement l'ensemble de `ACTIVE` tâches, car Amazon ECS ne gère plus le déploiement. Pour plus d'informations sur la suppression d'un ensemble de tâches, consultez le DeleteTaskSetmanuel Amazon Elastic Container Service API Reference.

Nombre de tâches souhaité

Le nombre d'instanciations de la tâche à placer et à poursuivre dans votre service.

Si vous souhaitez arrêter temporairement votre service, définissez cette valeur sur 0. Ensuite, lorsque vous êtes prêt à démarrer le service, mettez-le à jour avec la valeur d'origine.

Vous pouvez modifier ce paramètre pour les déploiements progressifs et les déploiements bleu/vert.

Activer les balises gérées

Détermine s'il faut activer les balises gérées Amazon ECS pour les tâches du service.

Seules les tâches lancées après la mise à jour refléteront la mise à jour. Pour mettre à jour les balises de toutes les tâches, utilisez l'option de déploiement forcé.

Vous pouvez modifier ce paramètre pour les déploiements progressifs et les déploiements bleu/vert.

Activer ECS Exec

Détermine si Amazon ECS Exec est utilisé.

Si vous ne souhaitez pas remplacer la valeur définie lors de la création du service, vous pouvez la définir sur null lors de l'exécution de cette action.

Vous pouvez modifier ce paramètre pour les déploiements progressifs.

Période de grâce pour le bilan de santé

Durée, en secondes, pendant laquelle le planificateur de services Amazon ECS ignore les vérifications incorrectes d'Elastic Load Balancing, de VPC Lattice et de l'état des conteneurs après le démarrage d'une tâche. Si vous ne spécifiez pas de valeur de période de grâce pour le bilan de santé, la valeur par défaut de `0` est utilisée. Si vous n'utilisez aucun des bilans de santé, cela signifie qu'il `healthCheckGracePeriodSeconds` n'est pas utilisé.

Si les tâches de votre service mettent un certain temps à démarrer et à répondre aux tests de santé, vous pouvez spécifier une période de grâce maximale de 2 147 483 647 secondes (environ 69 ans). Pendant ce temps, le planificateur du service Amazon ECS ignore le statut de la vérification de l'état. La période de grâce peut empêcher le planificateur de service de marquer les tâches comme étant non saines et de les arrêter avant qu'elles aient le temps de s'exécuter.

Vous pouvez modifier ce paramètre pour les déploiements progressifs et les déploiements bleu/vert.

Équilibreurs de charge

Vous devez utiliser un rôle lié à un service lorsque vous mettez à jour un équilibreur de charge.

Liste des objets de l'équilibreur de charge Elastic Load Balancing. Il contient le nom de l'équilibreur de charge, le nom du conteneur et le port du conteneur auquel accéder depuis l'équilibreur de charge. Le nom du conteneur est tel qu'il apparaît dans une définition de conteneur.

Amazon ECS ne met pas automatiquement à jour les groupes de sécurité associés aux équilibreurs de charge Elastic Load Balancing ou aux instances de conteneur Amazon ECS.

Lorsque vous ajoutez, mettez à jour ou supprimez une configuration d'équilibreur de charge, Amazon ECS démarre de nouvelles tâches avec la configuration Elastic Load Balancing mise à jour, puis arrête les anciennes tâches lorsque les nouvelles tâches sont en cours d'exécution.

Pour les services qui utilisent des mises à jour continues, vous pouvez ajouter, mettre à jour ou supprimer des groupes cibles Elastic Load Balancing. Vous pouvez passer d'un groupe cible unique à plusieurs groupes cibles et de plusieurs groupes cibles à un seul groupe cible.

Pour les services qui utilisent blue/green des déploiements, vous pouvez mettre à jour les groupes cibles d'Elastic Load Balancing en utilisant `CreateDeployment` through CodeDeploy. Notez que les groupes cibles multiples ne sont pas pris en charge pour les blue/green déploiements. Pour plus d'informations, voir Enregistrer plusieurs groupes cibles auprès d'un service.

Pour les services qui utilisent le contrôleur de déploiement externe, vous pouvez ajouter, mettre à jour ou supprimer des équilibreurs de charge en utilisant CreateTaskSet. Notez que les groupes cibles multiples ne sont pas pris en charge pour les déploiements externes. Pour plus d'informations, voir Enregistrer plusieurs groupes cibles auprès d'un service.

Passez une liste vide pour supprimer les équilibreurs de charge.

Vous pouvez modifier ce paramètre pour les déploiements progressifs.

Configuration réseau

Configuration du réseau de service.

Vous pouvez modifier ce paramètre pour les déploiements progressifs.

Contraintes de placement

Tableau d'objets de contrainte de placement de tâches permettant de mettre à jour le service à utiliser. Si aucune valeur n'est spécifiée, les contraintes de placement existantes pour le service resteront inchangées. Si cette valeur est spécifiée, elle remplacera toutes les contraintes de placement existantes définies pour le service. Pour supprimer toutes les contraintes de placement existantes, spécifiez un tableau vide.

Vous pouvez spécifier un maximum de 10 contraintes par tâche. Cette limite inclut les contraintes de la définition de tâche et celles spécifiées lors de l'exécution.

Vous pouvez modifier ce paramètre pour les déploiements progressifs et les déploiements bleu/vert.

Stratégie de placement

La stratégie de placement des tâches vise à mettre à jour le service à utiliser. Si aucune valeur n'est spécifiée, la stratégie de placement existante pour le service restera inchangée. Si cette valeur est spécifiée, elle remplacera la stratégie de placement existante définie pour le service. Pour supprimer une stratégie de placement existante, spécifiez un objet vide.

Vous pouvez modifier ce paramètre pour les déploiements progressifs et les déploiements bleu/vert.

Version de la plateforme

La version de la plateforme Fargate sur laquelle votre service s'exécute.

Un service utilisant une version de plateforme Linux ne peut pas être mis à jour pour utiliser une version de plateforme Windows et vice versa.

Vous pouvez modifier ce paramètre pour les déploiements progressifs.

Propager les tags

Détermine s'il faut propager les balises de la définition de tâche ou du service vers la tâche. Si aucune valeur n'est spécifiée, les balises ne sont pas propagées.

Seules les tâches lancées après la mise à jour refléteront la mise à jour. Pour mettre à jour les balises de toutes les tâches, définissez cette valeur sur`true`, `forceNewDeployment` afin qu'Amazon ECS lance de nouvelles tâches avec les balises mises à jour.

Vous pouvez modifier ce paramètre pour les déploiements progressifs et les déploiements bleu/vert.

Configuration de Service Connect

Configuration d'Amazon ECS Service Connect. Ce paramètre détermine la manière dont le service se connecte aux autres services de votre application.

Vous pouvez modifier ce paramètre pour les déploiements progressifs.

Registres de services

Vous devez utiliser un rôle lié à un service lorsque vous mettez à jour les registres de services.

Informations relatives aux registres de découverte de services à attribuer à ce service. Pour de plus amples informations, veuillez consulter Découverte de service.

Lorsque vous ajoutez, mettez à jour ou supprimez la configuration des registres de services, Amazon ECS lance de nouvelles tâches avec la configuration des registres de services mise à jour, puis arrête les anciennes tâches lorsque les nouvelles tâches sont en cours d'exécution.

Passez une liste vide pour supprimer les registres de services.

Vous pouvez modifier ce paramètre pour les déploiements progressifs.

Définition de la tâche

Définition et révision de la tâche à utiliser pour le service.

Si vous modifiez les ports utilisés par les conteneurs dans une définition de tâche, vous devrez peut-être mettre à jour les groupes de sécurité pour que les instances de conteneur fonctionnent avec les ports mis à jour.

Si vous mettez à jour la définition de tâche pour le service, le nom et le port de conteneur spécifiés dans la configuration de l'équilibreur de charge doivent rester dans la définition de tâche.

Le comportement d'extraction de l'image du conteneur diffère selon les types de lancement. Pour plus d'informations, consultez les étapes suivantes :

- Type de lancement Fargate pour Amazon ECS
- EC2 type de lancement pour Amazon ECS
- Type de lancement externe (Amazon ECS Anywhere) pour Amazon ECS

Vous pouvez modifier ce paramètre pour les déploiements progressifs.

Configuration du volume

Les détails du volume qui était`configuredAtLaunch`. Lorsqu'il `configuredAtLaunch` est défini sur `true` dans la définition de la tâche, ce paramètre de service configure un volume Amazon EBS pour chaque tâche du service à créer et à associer lors du déploiement. Vous pouvez configurer la taille, le type de volume, les IOPS, le débit, le snapshot et le chiffrement dans Configuration. ServiceManaged EBSVolume Le `name` volume doit correspondre à celui indiqué dans la définition `name` de la tâche. Si ce paramètre est défini sur null, aucun nouveau déploiement n'est déclenché. Dans le cas contraire, si cette configuration est différente de la configuration existante, elle déclenche un nouveau déploiement.

Vous pouvez modifier ce paramètre pour les déploiements progressifs.

Configuration du réseau VPC

La configuration du réseau VPC pour votre service. Cela définit la manière dont votre service s'intègre à VPC Lattice pour la communication. service-to-service

Vous pouvez modifier ce paramètre pour les déploiements progressifs.

AWS CDK considérations

Le AWS CDK ne suit pas l'état des ressources. Il ne sait pas si vous créez ou mettez à jour un service. Les clients doivent utiliser la trappe d'évacuation pour accéder directement à la construction ecs `Service` L1.

Pour plus d'informations sur les trappes d'échappement, voir Personnaliser les constructions à partir de la bibliothèque de constructions AWS dans le guide du développeur de la AWS Cloud Development Kit (AWS CDK) version 2.

Pour migrer votre service existant vers la `ecs.Service` construction, procédez comme suit :

1. Utilisez la trappe d'évacuation pour accéder à la construction `Service` L1.
2. Définissez manuellement les propriétés suivantes dans la construction `Service` L1.

   Si votre service utilise les EC2 capacités d'Amazon :

   - `daemon?`
   - `placementConstraints?`
   - `placementStrategies?`
   - Si vous utilisez le mode `awsvpc` réseau, vous devez définir les `securityGroups?` constructions `vpcSubnets?` et.

   Si votre service utilise Fargate :

   - `FargatePlatformVersion`
   - Les `vpcSubnets?` et les `securityGroups?` constructions.
3. Réglez le `launchType` comme suit :

```
const cfnEcsService = service.node.findChild('Service') as ecs.CfnService;
cfnEcsService.launchType = "FARGATE";
```

Pour passer d'un type de lancement à un fournisseur de capacité, procédez comme suit :

1. Utilisez la trappe d'évacuation pour accéder à la construction `Service` L1.
2. Ajoutez la `capacityProviderStrategies?` construction.
3. Déployez le service.

Mettre à jour un service Amazon ECS

Après avoir créé un service, il peut arriver que vous deviez mettre à jour les paramètres du service, par exemple le nombre de tâches.

Lorsque vous mettez à jour un service qui utilise un disjoncteur Amazon ECS, Amazon ECS crée un déploiement et une révision du service. Ces ressources vous permettent de consulter des informations détaillées sur l'historique des services. Pour de plus amples informations, veuillez consulter Afficher l'historique des services à l'aide des déploiements de services Amazon ECS.

Prérequis

Avant de mettre à jour un service, vérifiez quels paramètres de service peuvent être modifiés en fonction de votre type de déploiement. Pour obtenir la liste complète des paramètres modifiables, consultezMettre à jour les paramètres du service Amazon ECS.

Procédure

Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, cochez la case à côté du service, puis choisissez Mettre à jour.
4. Pour que votre service démarre un nouveau déploiement, sélectionnez Force new deployment (Forcer un nouveau déploiement).
5. Pour Définition de tâche, choisissez la famille et la révision de définition de tâche.

> **Important**
>
> La console vérifie que la famille de définitions de tâches et la révision sélectionnées sont compatibles avec la configuration de calcul définie. Si vous recevez un

avertissement, vérifiez à la fois la compatibilité de votre définition de tâche et la configuration de calcul que vous avez sélectionnée.

6. Si vous choisissez Replica pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer et à conserver dans le service.
7. Si vous avez choisi Replica, pour qu'Amazon ECS surveille la répartition des tâches entre les zones de disponibilité et les redistribue en cas de déséquilibre, sous Rééquilibrage du service des zones de disponibilité, sélectionnez Rééquilibrage du service des zones de disponibilité.
8. Pour Min running tasks (Minimum de tâches en cours d'exécution), saisissez la limite inférieure pour le nombre de tâches du service qui doivent rester à l'état `RUNNING` lors d'un déploiement, en tant que pourcentage de nombre souhaité de tâches (arrondi au nombre entier supérieur le plus proche). Pour plus d'informations, consultez Deployment configuration (Configuration de déploiement).
9. Pour Max running tasks (Maximum de tâches en cours d'exécution), saisissez la limite supérieure pour le nombre de tâches du service qui peuvent rester à l'état `RUNNING` ou `PENDING` lors d'un déploiement, en tant que pourcentage de nombre souhaité de tâches (arrondi au nombre entier inférieur le plus proche).
10. Pour configurer la manière dont les tâches sont déployées pour votre service, développez les options de déploiement, puis configurez vos options.

    a. Pour le type de contrôleur de déploiement, spécifiez le contrôleur de déploiement du service. La console Amazon ECS prend en charge les types de contrôleurs suivants :`ECS`.

    b. Pour Stratégie de déploiement, choisissez la stratégie utilisée par Amazon ECS pour déployer les nouvelles versions du service.

    c. En fonction de la stratégie de déploiement choisie, procédez comme suit :

Stratégie de déploiement	Étapes	
Mise à jour continue	a. Pour le nombre minimal de tâches en cours d'exécution %, spécifiez un pourcentage minimum de tâches qui doivent être exécutées lors du déploiement	

Stratégie de déploiement	Étapes	
	d'un service. Pour de plus amples informations, veuillez consulter Déployez les services Amazon ECS en remplaçant les tâches. b. Pour le nombre maximum de tâches en cours d'exécution, spécifiez une valeur maximale en pourcentage de tâches pouvant être exécutées lors du déploiement d'un service. Pour de plus amples informations, veuillez consulter Déployez les services Amazon ECS en remplaçant les tâches.	
Bleu/vert	Pour le temps de cuisson, spécifiez la durée, en minutes, pendant laquelle les révisions du service bleu et vert doivent être exécutées simultanément. Pour de plus amples informations, veuillez consulter blue/green Déploiements Amazon ECS.	

d. Pour exécuter des fonctions Lambda pendant une étape du cycle de vie, sous Deployment lifecyce hooks, effectuez les opérations suivantes pour chaque fonction Lambda unique :

   i. Choisissez Ajouter.

      Répétez l'opération pour chaque fonction unique que vous souhaitez exécuter.

   ii. Pour la fonction Lambda, entrez le nom de la fonction.

   iii. Pour Rôle, choisissez le rôle que vous avez créé dans les prérequis avec les blue/green autorisations.

      Pour de plus amples informations, veuillez consulter Autorisations requises pour les fonctions Lambda dans les déploiements Amazon ECS blue/green .

   iv. Pour les étapes du cycle de vie, sélectionnez les étapes exécutées par la fonction Lambda.

   v. (Facultatif) Pour les détails du crochet, entrez une paire clé-valeur fournissant des informations sur le crochet.

11. Pour configurer la manière dont Amazon ECS détecte et gère les échecs de déploiement, développez Deployment failure detection (Détection des échecs de déploiement), puis choisissez vos options.

    a. Pour arrêter un déploiement lorsque les tâches ne peuvent pas démarrer, sélectionnez Use the Amazon ECS deployment circuit breaker (Utiliser le disjoncteur de déploiement Amazon ECS).

       Pour que le logiciel annule automatiquement le déploiement au dernier état de déploiement terminé lorsque le disjoncteur de déploiement met le déploiement en état d'échec, sélectionnez Annulation en cas d'échec.

    b. Pour arrêter un déploiement en fonction des métriques de l'application, sélectionnez Utiliser une ou plusieurs CloudWatch alarmes. Ensuite, à partir du nom de l'CloudWatch alarme, choisissez les alarmes. Pour créer une nouvelle alarme, rendez-vous sur la CloudWatch console.

       Pour que le logiciel annule automatiquement le déploiement au dernier état de déploiement terminé lorsqu'une CloudWatch alarme indique que le déploiement a échoué, sélectionnez Annulation en cas d'échec.

12. Pour modifier les options de calcul, développez la configuration de calcul, puis procédez comme suit :

    a. Pour les services sur AWS Fargate, pour la version de la plateforme, choisissez la nouvelle version.

    b. Pour les services qui utilisent une stratégie de fournisseur de capacité, pour la stratégie de fournisseur de capacité, procédez comme suit :

    - Pour ajouter un fournisseur de capacité supplémentaire, choisissez Ajouter plus. Ensuite, pour Fournisseur de capacité, choisissez le fournisseur de capacité.
    - Pour supprimer un fournisseur de capacité, à droite du fournisseur de capacité, choisissez Supprimer.

    Un service qui utilise un fournisseur de capacité de groupe Auto Scaling ne peut pas être mis à jour pour utiliser un fournisseur de capacité Fargate. Un service qui utilise un fournisseur de capacité Fargate ne peut pas être mis à jour pour utiliser un fournisseur de capacité de groupe Auto Scaling.

13. (Facultatif) Pour configurer le service Auto Scaling, développez Service auto scaling, puis spécifiez les paramètres suivants. Pour utiliser Predicte Auto Scaling, qui examine les données de charge passées issues des flux de trafic, configurez-le après avoir créé le service. Pour de plus amples informations, veuillez consulter Utilisez des modèles historiques pour faire évoluer les services Amazon ECS grâce à une mise à l'échelle prédictive.

    a. Pour utiliser la mise à l'échelle automatique du service, sélectionnez Service Auto Scaling (mise à l'échelle automatique du service).

    b. Dans le champ Nombre minimum de tâches, entrez la limite inférieure du nombre de tâches à utiliser pour le dimensionnement automatique du service. Le nombre souhaité ne sera pas inférieur à ce nombre.

    c. Pour Nombre maximal de tâches, entrez la limite supérieure du nombre de tâches à utiliser pour le dimensionnement automatique du service. Le nombre souhaité ne sera pas supérieur à ce nombre.

    d. Choisissez le type de stratégie. Sous Type de politique de dimensionnement, choisissez l'une des options suivantes.

Pour utiliser ce type de politique	Faites ceci	
Suivi de la cible	a. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Target tracking (Suivi de cible). b. Pour Policy name (Nom de la politique), saisissez un nom de politique. c. Pour Métrique de service ECS, sélection nez l'une des métriques suivantes. • ECSServiceMoyenne CPUUtilization : utilisation moyenne du processeur par le service. • ECSServic eAverageMemoryUtil ization— Utilisati on moyenne de la mémoire du service. • ALBReques tCountPerTarget— Nombre de demandes traitées par cible dans un groupe cible d'Application Load Balancer.	

Pour utiliser ce type de politique	Faites ceci	
	d. PourTarget value (Valeur cible), entrez la valeur conservée par le service pour la métrique sélectionnée. e. Pour la période de recharge, entrez le temps, en secondes, après une activité de scale-out (ajout de tâches) qui doit s'écouler avant qu'une autre activité de scale-out puisse démarrer. f. Pour la période de recharge progressive, entrez le temps, en secondes, après une activité de mise à l'échelle (suppression de tâches) qui doit s'écouler avant qu'une autre activité de mise à l'échelle ne puisse démarrer. g. Pour empêcher la politique d'effectuer une activité de mise à l'échelle horizontale, sélectionnez Turn off scale-in (Désactiver la mise à l'échelle horizontale).	

Pour utiliser ce type de politique	Faites ceci	
	h. • (Facultatif) Sélectionnez Désactiver la mise à l'échelle si vous souhaitez que votre politique de dimensionnement soit adaptée à l'augmentation du trafic, mais que vous n'avez pas besoin qu'elle s'adapte lorsque le trafic diminue.	

Pour utiliser ce type de politique	Faites ceci	
Mise à l'échelle par étapes	a. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Mise à l'échelle par étapes. b. Pour Nom de la stratégie, saisissez un nom de stratégie. c. Pour Alarm name (Nom de l'alarme), entrez un nom unique pour l'alarme. d. Pour Métrique de service Amazon ECS, sélectionnez la métrique à utiliser pour l'alarme. e. Pour Statistique, choisissez la statistique d'alarme. f. Pour Période, choisisse z la période de l'alarme. g. Pour Condition d'alarme, choisissez comment comparer la métrique sélectionnée au seuil défini. h. Pour Seuil de comparaison des métriques et Période d'évaluation pour déclencher l'alarme,	

Pour utiliser ce type de politique	Faites ceci	
	saisissez le seuil utilisé pour l'alarme et la durée d'évaluation du seuil. i. Sous Actions de mise à l'échelle, procédez comme suit : • Pour Action, indiquez si vous souhaitez ajouter, supprimer ou définir un nombre spécifique souhaité pour votre service. • Si vous avez choisi d'ajouter ou de supprimer des tâches, dans Valeur, entrez le nombre de tâches (ou le pourcentage de tâches existante s) à ajouter ou à supprimer lorsque l'action de dimension nement est lancée. Si vous avez choisi de spécifier le nombre souhaité, saisissez le nombre de tâches. Pour Type, indiquez si la Valeur est un entier ou un pourcentage du	

Pour utiliser ce type de politique	Faites ceci	
	nombre souhaité existant. • Pour Limite inférieure et Limite supérieure, saisissez les limites inférieure et supérieure de votre ajustement de mise à l'échelle par étapes. Par défaut, la limite inférieure pour l'ajout d'une politique est le seuil de l'alarme et la limite supérieure est l'infini positif (+). Par défaut, la limite supérieure pour la suppression d'une politique est le seuil de l'alarme et la limite inférieure est l'infini négatif (-). • (Facultatif) Ajoutez des options de mise à l'échelle supplémentaires. Choisissez Ajouter une nouvelle action de dimensionnement, puis répétez les étapes des actions de dimensionnement.	

Pour utiliser ce type de politique	Faites ceci	
	• Pour la période de recharge, entrez le temps, en secondes, nécessaire pour qu'une activité de dimensionnement précédente prenne effet. Dans le cas d'une politique d'ajout, il s'agit du moment où, après une activité de scale-out, la politique de scale-out bloque les activités de scale-in et limite le nombre de tâches pouvant être redimensionnées à la fois. Pour une politique de suppression, il s'agit de la période qui suit une activité d'extension qui doit se terminer avant qu'une autre activité d'extension ne puisse démarrer.	

14. (Facultatif) Pour utiliser Service Connect, sélectionnez Turn on Service Connect (Activer Service Connect), puis spécifiez les informations suivantes :

    a. Sous Service Connect configuration (Configuration de Service Connect), spécifiez le mode client.

- Si votre service exécute une application client réseau qui doit uniquement se connecter aux autres services de l'espace de noms, choisissez Client side only.
- Si votre service exécute une application réseau ou un service Web et doit fournir des points de terminaison pour ce service, et se connecte à d'autres services dans l'espace de noms, choisissez Client and server (Client et serveur).

b. Pour utiliser un espace de noms qui n'est pas l'espace de noms de cluster par défaut, dans Namespace (Espace de noms), choisissez l'espace de noms du service.

15. Si votre tâche utilise un volume de données compatible avec la configuration lors du déploiement, vous pouvez configurer le volume en développant Volume.

    Le nom et le type de volume sont configurés lorsque vous créez une révision de définition de tâche et ne peuvent pas être modifiés lorsque vous mettez à jour un service. Pour mettre à jour le nom et le type du volume, vous devez créer une nouvelle révision de définition de tâche et mettre à jour le service en utilisant la nouvelle révision.

Pour configurer ce type de volume	Faites ceci	
Amazon EBS	a. Pour le type de volume EBS, choisissez le type de volume EBS que vous souhaitez associer à votre tâche. b. Pour Taille (GiB), entrez une valeur valide pour la taille du volume en gibioctets (GiB). Vous pouvez spécifier une taille de volume minimale de 1 GiB et maximale de 16 384 GiB. Cette valeur est obligatoire sauf si vous fournissez un identifiant de capture d'écran.	

Pour configurer ce type de volume	Faites ceci	
	c. Pour les IOPS, entrez le nombre maximum d' input/output opération s (IOPS) que le volume doit fournir. Cette valeur est configurable uniquement pour les types de `gp3` volume `io1``io2`, et. d. Pour Débit (Mib/s), entrez le débit que le volume doit fournir, en mégaoctets par seconde (ou Mib/s). MiBps Cette valeur est configurable uniquement pour le type de `gp3` volume. e. Pour Snapshot ID, choisissez un instantan é de volume Amazon EBS existant ou entrez l'ARN d'un instantané si vous souhaitez créer un volume à partir d'un instantané. Vous pouvez également créer un nouveau volume vide en ne choisissant ni en saisissant un identifiant de capture d'écran. f. Si vous spécifiez un ID de capture d'écran, vous pouvez spécifier un taux	

Pour configurer ce type de volume	Faites ceci	
	d'initialisation du volume (Mbits/s). Entrez une valeur comprise entre 100 et 300, en Mbits/s, qui déterminera la vitesse de chargement des données à partir de l'instantané spécifié à l'aide de l'ID de capture pour la création du volume. g. Pour Type de système de fichiers, choisissez le type de système de fichiers qui sera utilisé pour le stockage et la récupération des données sur le volume. Vous pouvez choisir le système d'exploitation par défaut ou un type de système de fichiers spécifique. La valeur par défaut pour Linux est`XFS`. Pour les volumes créés à partir d'un instantané, vous devez spécifier le même type de système de fichiers que celui utilisé par le volume lors de la création de l'instantané. Si le type de système de fichiers ne	

Pour configurer ce type de volume	Faites ceci	
	correspond pas, la tâche ne démarrera pas. h. Pour le rôle d'infrastructure, choisissez un rôle IAM doté des autorisations nécessaires pour permettre à Amazon ECS de gérer les volumes Amazon EBS pour les tâches. Vous pouvez associer la politique `AmazonECSInfrastructureRolePolicyForVolumes` gérée au rôle, ou vous pouvez utiliser la politique comme guide pour créer et associer votre propre politique avec des autorisations répondant à vos besoins spécifiques. Pour plus d'informations sur les autorisations nécessaires, consultez Rôle IAM dans l'infrastructure Amazon ECS. i. Pour le chiffrement, choisissez Par défaut si vous souhaitez utiliser le chiffrement Amazon EBS par défaut.	

Pour configurer ce type de volume	Faites ceci	
	Si le chiffrement est configuré par défaut sur votre compte, le volume sera chiffré avec la clé AWS Key Management Service (AWS KMS) spécifiée dans le paramètre. Si vous choisissez Par défaut et que le chiffrement par défaut d'Amazon EBS n'est pas activé, le volume ne sera pas chiffré. Si vous choisissez Personnalisé, vous pouvez spécifier celui AWS KMS key de votre choix pour le chiffrement du volume. Si vous choisissez Aucun, le volume ne sera pas chiffré, sauf si le chiffrement est configuré par défaut ou si vous créez un volume à partir d'un instantané chiffré. j. Si vous avez choisi Personnalisé pour le chiffrement, vous devez spécifier celui AWS KMS key que vous souhaitez	

Pour configurer ce type de volume	Faites ceci	
	utiliser. Pour la clé KMS, choisissez AWS KMS key ou entrez un ARN de clé. Si vous choisissez de chiffrer votre volume à l'aide d'une clé symétrique gérée par le client, assurez-vous que vous disposez des autorisations appropriées définies dans votre AWS KMS key politique. Pour plus d'informations, consultez la section Chiffrement des données pour les volumes Amazon EBS. k. (Facultatif) Sous Balises, vous pouvez ajouter des balises à votre volume Amazon EBS soit en propageant des balises à partir de la définition de la tâche ou du service, soit en fournissant vos propres balises. Si vous souhaitez propager des balises à partir de la définition de tâche, choisissez Définition de tâche pour propager des balises à partir de. Si vous souhaitez propager des	

Pour configurer ce type de volume	Faites ceci	
	balises depuis le service, choisissez Service for Propagate tags from. Si vous choisissez Ne pas propager, ou si vous ne choisissez aucune valeur, les balises ne sont pas propagées. Si vous souhaitez fournir vos propres balises, choisissez Ajouter une balise, puis indiquez la clé et la valeur pour chaque balise que vous ajoutez. Pour plus d'informations sur le balisage des volumes Amazon EBS, consultez la section Marquage des volumes Amazon EBS.	

16. (Facultatif) Pour vous aider à identifier votre service, développez Tags (Balises), puis configurez vos balises.
    - [Ajouter un tag] Choisissez Ajouter un tag, puis procédez comme suit :
        - Pour Clé, saisissez le nom de la clé.
        - Pour Valeur, saisissez la valeur de clé.
    - [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.
17. Choisissez Mettre à jour.

AWS CLI

- Exécutez `update-service`. Pour plus d'informations sur l'exécution de la commande, voir update-service dans la AWS Command Line Interface référence.

  L'`update-service`exemple suivant met à jour le nombre de tâches souhaité du service `my-http-service` à 2.

  Remplacez le *`user-input`* par vos valeurs.

  ```
  aws ecs update-service \
      --cluster MyCluster \
      --service my-http-service \
      --desired-count 2
  ```

Étapes suivantes

Suivez votre déploiement et consultez l'historique de vos services pour les services qu'Amazon ECS disjoncte. Pour de plus amples informations, veuillez consulter Afficher l'historique des services à l'aide des déploiements de services Amazon ECS.

Supprimer un service Amazon ECS à l'aide de la console

Voici quelques-unes des raisons pour lesquelles vous voudriez supprimer un service :

- L'application n'est plus nécessaire
- Vous migrez le service vers un nouvel environnement
- L'application n'est pas utilisée activement
- L'application utilise plus de ressources que nécessaire et vous essayez d'optimiser vos coûts

La capacité du service est automatiquement réduite à 0. Les ressources de l'équilibreur de charge ou de la découverte de service qui sont associées au service, elles ne sont pas affectées par la suppression de service. Pour supprimer vos ressources Elastic Load Balancing, consultez l'une des rubriques suivantes, en fonction du type de votre équilibreur de charge : Suppression d'un Application Load Balancer ou Suppression d'un Network Load Balancer.

Lorsque vous supprimez un service, Amazon ECS supprime tous les déploiements de services et toutes les révisions de service pour le service.

Lorsque vous supprimez un service, si des tâches de nettoyage sont toujours en cours d'exécution, le statut du service passe de `ACTIVE` à `DRAINING` et le service n'est plus visible dans la console ou dans le fonctionnement de l'`ListServices`API. Une fois que toutes les tâches sont passées à l'état `STOPPING` ou `STOPPED`, l'état du service passe de `DRAINING` à `INACTIVE`. Les services ayant le `INACTIVE` statut `DRAINING` ou peuvent toujours être visualisés avec l'opération de l'`DescribeServices`API.

> **Important**
>
> Si vous tentez de créer un nouveau service portant le même nom qu'un service existant dans l'un `ACTIVE` ou l'autre `DRAINING` statut, vous recevrez un message d'erreur.

Procédure

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, sélectionnez le cluster pour le service.
3. Sur la page Clusters, choisissez le cluster.
4. Sur la *name* page Cluster :, choisissez l'onglet Services.
5. Sélectionnez les services, puis choisissez Delete (Supprimer).
6. Pour supprimer un service même s'il n'a pas été réduit à zéro tâche, sélectionnezForce delete service (Forcer la suppression du service).
7. À l'invite de confirmation, saisissez Supprimer, puis sélectionnez Supprimer.

Migrer un ARN de service court Amazon ECS vers un ARN long

Amazon ECS attribue un Amazon Resource Name (ARN) unique à chaque service. Les services créés avant 2021 ont un format ARN court :

```
arn:aws:ecs:region:aws_account_id:service/service-name
```

Amazon ECS a modifié le format ARN pour inclure le nom du cluster. Il s'agit d'un format d'ARN long :

```
arn:aws:ecs:region:aws_account_id:service/cluster-name/service-name
```

Votre service doit avoir le format ARN long pour pouvoir le baliser.

Vous pouvez migrer un service au format ARN court vers le format ARN long sans avoir à recréer le service. Vous pouvez utiliser l'API, la CLI ou la console. Vous ne pouvez pas annuler l'opération de migration.

Le processus de migration est fluide et garantit l'absence de temps d'arrêt pour votre service. Pendant la migration :

- Disponibilité du service : votre service continue de fonctionner normalement sans interruption du trafic ou des fonctionnalités.
- Tâches en cours : les tâches existantes continuent de s'exécuter sans interruption. Les nouvelles tâches lancées après la migration utiliseront le format ARN long si le paramètre du `taskLongArnFormat` compte est activé.
- Instances de conteneur : les instances de conteneur ne sont pas affectées par la migration de l'ARN du service et continuent de fonctionner normalement.
- Configuration du service : Tous les paramètres du service, y compris la définition des tâches, le réseau et les configurations de l'équilibreur de charge, restent inchangés.

Si vous souhaitez l'utiliser AWS CloudFormation pour étiqueter un service au format ARN court, vous devez migrer le service à l'aide de l'API, de la CLI ou de la console. Une fois la migration terminée, vous pouvez l'utiliser AWS CloudFormation pour étiqueter le service.

Si vous souhaitez utiliser Terraform pour étiqueter un service au format ARN court, vous devez migrer le service à l'aide de l'API, de la CLI ou de la console. Une fois la migration terminée, vous pouvez utiliser Terraform pour étiqueter le service.

Une fois la migration terminée, le service apporte les modifications suivantes :

- Le format ARN long

  `arn:aws:ecs:region:aws_account_id:service/cluster-name/service-name`

- Lorsque vous effectuez une migration à l'aide de la console, Amazon ECS ajoute une balise au service dont la clé est définie sur « ecs : serviceArnMigrated At » et la valeur sur l'horodatage de la migration (format UTC).

  Ce tag est pris en compte dans votre quota de tags.

- Lorsque le `PhysicalResourceId` in a AWS CloudFormation stack représente un ARN de service, la valeur ne change pas et restera l'ARN du service court.

Prérequis

Effectuez les opérations suivantes avant de migrer l'ARN du service.

1. Pour savoir si vous avez un ARN de service court, consultez les détails du service dans la console Amazon ECS (un avertissement s'affiche lorsque le service utilise le format d'ARN court) ou le paramètre de `serviceARN` retour de`describe-services`. Lorsque l'ARN n'inclut pas le nom du cluster, vous disposez d'un ARN court. Le format d'un ARN court est le suivant :

   `arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`service-name`*

2. Notez la date de création.
3. Si vous avez des politiques IAM qui utilisent le format d'ARN court, mettez-le à jour au format d'ARN long.

   Remplacez chaque *`user input placeholder`* par vos propres informations.

   `arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`cluster-name`*`/`*`service-name`*

   Pour plus d'informations, consultez la section Modification des politiques IAM dans le guide de l' AWS Identity and Access Management utilisateur.
4. Si vous disposez d'outils qui utilisent le format d'ARN court, mettez-le à jour au format d'ARN long.

   Remplacez chaque *`user input placeholder`* par vos propres informations.

   `arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`cluster-name`*`/`*`service-name`*
5. Activez le format ARN long du service. Exécutez `put-account-setting` avec l'option `serviceLongArnFormat` définie à `enabled`. Pour plus d'informations, put-account-settingconsultez le manuel Amazon Elastic Container Service API Reference.

   Exécutez la commande en tant qu'utilisateur root lorsque la `createdAt` date de votre service est inconnue.

   ```
   aws ecs put-account-setting --name serviceLongArnFormat --value enabled
   ```

   Exemple de sortie

   ```
   {
       "setting": {
   ```

```
        "name": "serviceLongArnFormat",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:role/your-role",
        "type": user
    }
}
```

6. Activez le format ARN long de la tâche. Ce paramètre de compte contrôle le format ARN des nouvelles tâches lancées une fois la migration du service terminée. Exécutez `put-account-setting` avec l'option `taskLongArnFormat` définie à `enabled`. Pour plus d'informations, put-account-settingconsultez le manuel Amazon Elastic Container Service API Reference.

   Exécutez la commande en tant qu'utilisateur root lorsque la `createdAt` date de votre service est inconnue.

```
aws ecs put-account-setting --name taskLongArnFormat --value enabled
```

   Exemple de sortie

```
{
    "setting": {
        "name": "taskLongArnFormat",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:role/your-role",
        "type": user
    }
}
```

> **Note**
>
> Le `taskLongArnFormat` paramètre ne migre pas directement les tâches existantes. Cela n'affecte que le format ARN des nouvelles tâches créées après l'activation du paramètre. Les tâches en cours d'exécution existantes conservent leur format ARN actuel jusqu'à ce qu'elles soient remplacées dans le cadre d'opérations de service normales (telles que des déploiements ou des activités de dimensionnement).

Procédure

Utilisez ce qui suit pour migrer l'ARN de votre service.

console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster.
3. Dans la section Services, choisissez un service dont la colonne ARN contient un avertissement.

   La page des détails du service apparaît.
4. Choisissez Migrer vers un ARN long.

   La boîte de dialogue du service Migrate apparaît.
5. Choisissez Migrate (Migrer).

INTERFACE DE LIGNE DE COMMANDE (CLI)

Une fois les prérequis remplis, vous pouvez étiqueter votre service. Exécutez la commande suivante :

Amazon ECS envisage de transmettre le format d'ARN long dans une demande d'`tag-resource`API pour un service avec un ARN court comme signal de migration du service afin d'utiliser le format d'ARN long.

```
aws ecs tag-resource \
    --resource-arn arn:aws:ecs:region:aws_account_id:service/cluster-name/service-name
    --tags key=key1,value=value1
```

Voici quelques exemples de balises MyService dont la clé est définie sur « TestService » et la valeur sur « » WebServers :

```
aws ecs tag-resource \
    --resource-arn arn:aws:ecs:us-east-1:123456789012:service/MyCluster/MyService
    --tags key=TestService1,value=WebServers
```

Terraform

Une fois les prérequis remplis, vous pouvez étiqueter votre service. Créez une `aws_ecs_service` ressource et définissez la `tags` référence. Pour plus d'informations, consultez Resource : aws_ecs_service dans la documentation de Terraform.

```
resource "aws_ecs_service" "MyService" {
  name    = "example"
```

```
  cluster = aws_ecs_cluster.MyService.id

 tags = {
 "Name"  =  "MyService"
 "Environment"  =  "Production"
 "Department"  =  "QualityAssurance"
  }
}
```

Étapes suivantes

Vous pouvez ajouter des balises au service. Pour de plus amples informations, veuillez consulter Ajouter des balises aux ressources Amazon ECS.

Si vous souhaitez qu'Amazon ECS propage les balises de la définition de tâche ou du service vers la tâche, exécutez `update-service` le `propagateTags` paramètre. Pour plus d'informations, voir update-service dans la AWS Command Line Interface référence.

Résolution des problèmes

Certains utilisateurs peuvent rencontrer l'erreur suivante lorsqu'ils migrent du format d'ARN court vers le format d'ARN long.

`There was an error while migrating the ARN of service` *`service-name`*`. The specified account does not have serviceLongArnFormat or taskLongArnFormat account settings enabled. Add account settings in order to enable tagging.`

Si vous avez déjà activé les paramètres du `serviceLongArnFormat` compte mais que vous rencontrez toujours cette erreur, cela peut être dû au fait que les paramètres du compte pour le format ARN long n'ont pas été activés pour le principal IAM qui a initialement créé le service.

1. Identifiez le principal qui a créé le service.

   1. Dans la console, les informations sont disponibles dans le champ Créé par de l'onglet Configuration et mise en réseau de la page Détails du service de la console Amazon ECS.
   2. Pour le AWS CLI, exécutez la commande suivante :

      Remplacez le *`user-input`* par vos valeurs.

      ```
      aws ecs describe-services --cluster cluster-name --services service-name --query
       'services[0].{createdBy: createdBy}'
      ```

2. Activez les paramètres de compte requis pour ce principal spécifique. Vous pouvez effectuer cette opération de différentes manières :

   a. Assumez l'utilisateur ou le rôle IAM pour ce principal. Ensuite, exécutez `put-account-setting`.

   b. Utilisez l'utilisateur root pour exécuter la commande tout en spécifiant le principal de création à l'aide du`principal-arn`.

      Exemple.

      Remplacez le *`principal-arn`* par la valeur de l'étape 1.

      ```
      aws ecs put-account-setting --name serviceLongArnFormat --value enabled --
      principal-arn arn:aws:iam::123456789012:role/jdoe
      ```

Les deux méthodes activent le paramètre de `serviceLongArnFormat` compte requis sur le principal qui a créé le service, ce qui permet de poursuivre la migration de l'ARN.

Logique de régulation du service Amazon ECS

Le planificateur de service Amazon ECS inclut une logique de protection qui limite le lancement des tâches lorsque celles-ci échouent à plusieurs reprises. Cela permet d'éviter une consommation inutile de ressources et de réduire les coûts.

Lorsque les tâches d'un service ne passent pas d'un `RUNNING` état `PENDING` à un autre et passent directement à un état à un autre`STOPPED`, le planificateur :

- Augmente progressivement le délai entre les tentatives de redémarrage
- Continue d'augmenter les délais jusqu'à un maximum de 27 minutes entre les tentatives
- Génère un message d'événement de service pour vous informer du problème

> **Note**
>
> Le délai maximum de 27 minutes peut changer dans les futures mises à jour.

Lorsque la régulation est activée, vous recevez ce message d'événement de service :

```
(service service-name) is unable to consistently start tasks successfully.
```

Caractéristiques importantes de la logique de l'accélérateur :

- Les services continuent de réessayer indéfiniment
- La seule modification est l'augmentation du temps entre les redémarrages
- Il n'existe aucun paramètre configurable par l'utilisateur

Résolution des problèmes de régulation

Pour résoudre le problème de l'étranglement, vous pouvez :

- Mettez à jour le service pour utiliser une nouvelle définition de tâche, qui rétablit immédiatement le fonctionnement normal du service, sans limitation. Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.
- Corrigez la cause sous-jacente des échecs des tâches.

Les causes courantes d'échec des tâches qui déclenchent la régulation sont les suivantes :

- Ressources de cluster insuffisantes (ports, mémoire ou processeur)
  - Signalé par un message d'événement relatif à un service de ressources insuffisantes
- Défaillances lors de l'extraction des images
  - Cela peut être dû à des noms d'image non valides, à des balises ou à des autorisations insuffisantes
  - Résultats `CannotPullContainerError` en L'affichage d'Amazon ECS a permis de stopper les erreurs de tâches
- Insufficient disk space (Espace disque insuffisant)
  - Se traduit par des erreurs liées `CannotCreateContainerError` à l'arrêt des tâches
  - Pour les étapes de résolution, voir Résoudre les problèmes liés au Docker dans `API error (500): devmapper` Amazon ECS

> **Important**
>
> Les scénarios suivants ne déclenchent PAS la logique d'accélération :

- Tâches qui s'arrêtent après avoir atteint `RUNNING` l'état
- Tâches arrêtées en raison de l'échec des tests de santé d'Elastic Load Balancing
- Tâches pour lesquelles la commande du conteneur se termine avec un code différent de zéro après avoir atteint l'état `RUNNING`

Paramètres de définition du service Amazon ECS

Une définition de service définit comment exécuter votre service Amazon ECS service. Les paramètres suivants peuvent être spécifiés dans une définition de service.

Type de lancement

`launchType`

Type : String

Valeurs valides : `EC2` | `FARGATE` | `EXTERNAL`

Obligatoire : non

Type de lancement sur lequel vous pouvez exécuter votre service. Si aucun type de lancement n'est spécifié, the `capacityProviderStrategy` par défaut est utilisé par défaut. Pour de plus amples informations, veuillez consulter Types de lancement Amazon ECS.

Si un `launchType` est spécifié, le paramètre `capacityProviderStrategy` doit être omis.

Stratégie de fournisseur de capacité

`capacityProviderStrategy`

Type : tableau d'objets

Obligatoire : non

Stratégie du fournisseur de capacité à utiliser pour le service.

Une stratégie de fournisseur de capacité consiste en un ou plusieurs fournisseurs de capacité avec le `base` et `weight` à leur attribuer. Un fournisseur de capacité doit être associé au cluster à utiliser dans une stratégie de fournisseur de capacité. L' PutClusterCapacityProviders API est

utilisée pour associer un fournisseur de capacité à un cluster. Seuls les fournisseurs de capacité ayant un statut `ACTIVE` ou `UPDATING` peuvent être utilisés.

Si un `capacityProviderStrategy` est spécifié, le paramètre `launchType` doit être omis. Lorsque ni `capacityProviderStrategy` ni `launchType` ne sont spécifiés, le `defaultCapacityProviderStrategy` pour le cluster est utilisé.

Si vous souhaitez spécifier un fournisseur de capacité qui utilise un groupe Auto Scaling, le fournisseur de capacité doit déjà être créé. De nouveaux fournisseurs de capacité peuvent être créés à l'aide de l'opération CreateCapacityProvider API.

Pour utiliser un AWS fournisseur de capacité Fargate, spécifiez le ou les fournisseurs `FARGATE` de `FARGATE_SPOT` capacité. Les fournisseurs AWS de capacité Fargate sont disponibles pour tous les comptes et doivent uniquement être associés à un cluster pour être utilisés.

L'opération PutClusterCapacityProviders API est utilisée pour mettre à jour la liste des fournisseurs de capacité disponibles pour un cluster après sa création.

`capacityProvider`

Type : String

Obligatoire : oui

Le nom abrégé ou l'Amazon Resource Name (ARN) complet du fournisseur de capacité.

`weight`

Type : entier

Plage valide : entiers compris entre 0 et 1 000.

Obligatoire : non

La valeur de poids indique le pourcentage relatif du nombre total de tâches lancées qui utilisent le fournisseur de capacité spécifié.

Par exemple, supposons que vous avez une stratégie qui contient deux fournisseurs de capacité et que les deux ont un poids de un. Lorsque la base est satisfaite, les tâches sont réparties équitablement entre les deux fournisseurs de capacité. Dans la même logique, supposons que vous spécifiez un poids de 1 pour capacityProviderA et un poids de 4 pour capacityProviderB. Ensuite, pour chaque tâche exécutée avec capacityProviderA, quatre tâches utilisent capacityProviderB.

`base`

Type : entier

Plage valide : entiers compris entre 0 et 100 000.

Obligatoire : non

La valeur de base indique le nombre minimum de tâches à exécuter sur le fournisseur de capacité spécifié. Une base ne peut être définie que pour un seul fournisseur de capacité dans une stratégie de fournisseur de capacité.

Définition de tâche

`taskDefinition`

Type : chaîne

Obligatoire : non

L'Amazon Resource Name (ARN) complet ou `family` et `revision` (`family:revision`) de la définition de tâche à exécuter dans votre service. Si un `revision` n'est pas spécifié, la dernière révision `ACTIVE` de la famille spécifiée est utilisée.

Une définition de tâche doit être spécifiée lors de l'utilisation du contrôleur de déploiement de mise à jour continue (`ECS`).

Système d'exploitation de la plateforme

`platformFamily`

Type : chaîne

Obligatoire : Conditionnelle

Par défaut : Linux

Ce paramètre est requis pour les services Amazon ECS services hébergés sur Fargate.

Ce paramètre est ignoré pour les services Amazon ECS hébergés sur Amazon EC2.

Système d'exploitation des conteneurs qui exécutent le service. Les valeurs valides sont `LINUX`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE`, `WINDOWS_SERVER_2022_FULL` et `WINDOWS_SERVER_2022_CORE`.

La valeur `platformFamily` pour chaque tâche que vous spécifiez pour le service doit correspondre à la valeur `platformFamily` du service. Par exemple, si vous définissez `platformFamily` sur `WINDOWS_SERVER_2019_FULL`, la valeur `platformFamily` de toutes les tâches doit être `WINDOWS_SERVER_2019_FULL`.

Version de plateforme

`platformVersion`

Type : chaîne

Obligatoire : non

Version de la plateforme sur laquelle s'exécutent vos tâches dans le service. Une version de plateforme est spécifiée uniquement pour les tâches utilisant le type de lancement Fargate. Si vous ne spécifiez aucune valeur, la dernière version (`LATEST`) est utilisée par défaut.

AWS Les versions de la plate-forme Fargate sont utilisées pour faire référence à un environnement d'exécution spécifique pour l'infrastructure de tâches Fargate. Lorsque vous spécifiez `LATEST` comme version de plateforme lors de l'exécution d'une tâche ou de la création d'un service, vous obtenez la version de plateforme la plus récente pour vos tâches. Lorsque vous mettez à l'échelle votre service, ces tâches reçoivent la version de plateforme spécifiée dans le déploiement actuel du service. Pour de plus amples informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.

> **Note**
>
> Les versions de plateforme ne sont pas spécifiées pour les tâches utilisant le type de lancement EC2.

Cluster

`cluster`

Type : chaîne

Obligatoire : non

Nom court ou Amazon Resource Name (ARN) complet du cluster sur lequel exécuter votre service. Si vous ne spécifiez aucun cluster, le cluster `default` est sélectionné.

Nom du service

`serviceName`

Type : String

Obligatoire : oui

Nom de votre service. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres, des tirets ou des traits de soulignement. Dans un cluster, les noms de service doivent être uniques, mais certains services peuvent avoir des noms similaires dans des clusters différents d'une même région ou de plusieurs régions.

Stratégie de planification

`schedulingStrategy`

Type : String

Valeurs valides : `REPLICA` | `DAEMON`

Obligatoire : non

Stratégie de planification à utiliser. Si aucune stratégie de planification n'est spécifiée, la stratégie `REPLICA` est utilisée. Pour de plus amples informations, veuillez consulter Services Amazon ECS.

Deux stratégies de planificateur de service sont disponibles :

- `REPLICA` – La stratégie de planification des réplicas place et gère le nombre de tâches souhaité dans votre cluster. Par défaut, le planificateur de service répartit les tâches entre les zones de disponibilité. Vous pouvez utiliser des stratégies et des contraintes de placement des tâches afin de personnaliser la façon dont les tâches sont placées. Pour de plus amples informations, veuillez consulter Stratégie de planification des répliques.
- `DAEMON` – La stratégie de planification du démon déploie exactement une tâche sur chaque instance de conteneur active qui répond à toutes les contraintes de placement des tâches spécifiées dans votre cluster. Lors de l'utilisation de cette stratégie, il n'est pas nécessaire de

spécifier un nombre de tâches souhaité, une stratégie de placement des tâches ou d'utiliser les politiques Service Auto Scaling. Pour de plus amples informations, veuillez consulter Stratégie de planification des démons.

> **Note**
>
> Les tâches Fargate ne prennent pas en charge la stratégie de planification `DAEMON`.

Nombre souhaité

`desiredCount`

Type : entier

Obligatoire : non

Nombre d'instances de la définition de tâche spécifiée à placer et à exécuter dans votre service.

Ce paramètre est requis si la stratégie de planification `REPLICA` est utilisée. Si le service utilise la stratégie de planification `DAEMON`, ce paramètre est facultatif.

Lorsque vous utilisez le dimensionnement automatique des services, lorsque vous mettez à jour un service en cours d'exécution avec un nombre de tâches `desiredCount` inférieur au nombre de tâches en cours d'exécution, le service est redimensionné jusqu'au niveau spécifié`desiredCount`.

Configuration de déploiement

`deploymentConfiguration`

Type : objet

Obligatoire : non

Paramètres de déploiement facultatifs qui contrôlent le nombre de tâches exécutées durant le déploiement et la commande d'arrêt et de démarrage des tâches.

`maximumPercent`

Type : entier

Obligatoire : non

Si un service utilise le type de déploiement de mise à jour propagée (`ECS`), le paramètre `maximumPercent` représente une limite supérieure du nombre de tâches de votre service autorisées à avoir l'état `RUNNING`, `STOPPING` ou `PENDING` pendant un déploiement. Il est exprimé en tant que pourcentage de `desiredCount` arrondi à la valeur inférieure la plus proche. Vous pouvez utiliser ce paramètre pour définir la taille des lots de déploiement. Par exemple, si votre service utilise le planificateur de `REPLICA` services et comporte quatre tâches et une `maximumPercent` valeur `desiredCount` de 200 %, le planificateur démarre quatre nouvelles tâches avant d'arrêter les quatre tâches plus anciennes. Cela s'applique à condition que les ressources de cluster nécessaires à cette opération soient disponibles. La valeur `maximumPercent` par défaut pour un service à l'aide du planificateur de service `REPLICA` est de 200 %.

Le planificateur Amazon ECS utilise ce paramètre pour remplacer les tâches défectueuses en démarrant d'abord les tâches de remplacement, puis en arrêtant les tâches défectueuses, à condition que les ressources du cluster permettant de démarrer les tâches de remplacement soient disponibles. Pour plus d'informations sur la façon dont le planificateur remplace les tâches défectueuses, consultez. Services Amazon ECS

Si votre service utilise le type de planificateur de service `DAEMON`, le `maximumPercent` doit rester à 100 %. C'est la valeur par défaut.

Le nombre maximum de tâches lors d'un déploiement est le `desiredCount` multiplié par le `maximumPercent`/100, arrondi au nombre entier inférieur le plus proche.

Si un service utilise le type bleu/vert (`CODE_DEPLOY`) ou le type de `EXTERNAL` déploiement et les tâches utilisant le type de EC2 lancement, la valeur maximale en pourcentage est définie sur la valeur par défaut. La valeur est utilisée pour définir la limite supérieure du nombre de tâches du service qui restent dans `RUNNING` l'état alors que les instances de conteneur sont dans `DRAINING` cet état.

> **Note**
>
> Vous ne pouvez pas spécifier de `maximumPercent` valeur personnalisée pour un service qui utilise le type bleu/vert (`CODE_DEPLOY`) ou le type de `EXTERNAL` déploiement et dont les tâches utilisent le type de EC2 lancement.

Si le service utilise le type bleu/vert (`CODE_DEPLOY`) ou le type de `EXTERNAL` déploiement, et que les tâches du service utilisent le type de lancement Fargate, la valeur maximale en pourcentage n'est pas utilisée. La valeur est toujours renvoyée lors de la description de votre service.

`minimumHealthyPercent`

Type : entier

Obligatoire : non

Si un service utilise le type de déploiement de mise à jour propagée (`ECS`), `minimumHealthyPercent` représente une limite inférieure du nombre de tâches de votre service qui doivent rester l'état `RUNNING` pendant un déploiement. Cela est exprimé en tant que pourcentage de `desiredCount` arrondi à la valeur supérieure la plus proche. Vous pouvez utiliser ce paramètre pour procéder au déploiement sans avoir recours à une capacité de cluster supplémentaire.

Par exemple, si votre service comporte quatre tâches, une `desiredCount` de 50 % et une `minimumHealthyPercent` de 100 %, le planificateur `maximumPercent` de services arrête deux tâches existantes pour libérer de la capacité du cluster avant de démarrer deux nouvelles tâches.

Si des tâches ne fonctionnent pas correctement et s'il `maximumPercent` n'autorise pas le planificateur Amazon ECS à démarrer des tâches de remplacement, le planificateur arrête les tâches défectueuses one-by-one (en utilisant la `minimumHealthyPercent` contrainte) afin de libérer la capacité de lancer des tâches de remplacement. Pour plus d'informations sur la façon dont le planificateur remplace les tâches défectueuses, consultez. Services Amazon ECS

Pour les services qui n'utilisent pas d'équilibreur de charge, il convient de noter ce qui suit :

- Un service est considéré comme intègre si tous les conteneurs essentiels dans les tâches du service réussissent leurs surveillances d'état.
- Si une tâche n'a pas de conteneurs essentiels avec une surveillance de l'état définie, le planificateur de service attend 40 secondes une fois qu'une tâche a atteint l'état `RUNNING` avant que celle-ci soit comptée dans le pourcentage minimal total d'intégrité.
- Si une tâche comporte un ou plusieurs conteneurs essentiels avec une surveillance de l'état définie, le planificateur de service attend que la tâche atteigne un état intègre avant de la compter dans le pourcentage minimum total d'intégrité. Une tâche est considérée comme

intègre lorsque tous ses conteneurs essentiels ont réussi leurs surveillances d'état. La durée pendant laquelle le planificateur de service peut attendre est déterminée par les paramètres de surveillance de l'état du conteneur. Pour de plus amples informations, veuillez consulter Surveillance de l'état.

Pour les services qui utilisent un équilibreur de charge, il convient de noter ce qui suit :

- Si une tâche n'a pas de conteneurs essentiels avec une surveillance de l'état définie, le planificateur de services attend que la surveillance de l'état du groupe cible de l'équilibreur de charge retrouve un état intègre avant de compter la tâche dans le pourcentage minimal total d'intégrité.
- Si une tâche possède un conteneur essentiel avec une surveillance de l'état définie, le planificateur de services attend que la tâche atteigne un état intègre et que la surveillance de l'état du groupe cible d'équilibrage de charge retrouve un état intègre avant de compter la tâche dans le pourcentage minimal total d'intégrité.

La valeur par défaut d'un service de réplica pour `minimumHealthyPercent` est 100 %. La `minimumHealthyPercent` valeur par défaut pour un service utilisant le calendrier de `DAEMON` service est 0 % pour le AWS CLI AWS SDKs, le, APIs et 50 % pour le AWS Management Console.

Le nombre minimum de tâches saines lors d'un déploiement correspond à `desiredCount` multiplié par `minimumHealthyPercent`/100, arrondi au nombre entier supérieur le plus proche.

Si un service utilise le type bleu/vert (`CODE_DEPLOY`) ou le type de `EXTERNAL` déploiement et exécute des tâches utilisant le type de EC2 lancement, la valeur minimale du pourcentage sain est définie sur la valeur par défaut. La valeur est utilisée pour définir la limite inférieure du nombre de tâches du service qui restent dans `RUNNING` l'état alors que les instances de conteneur sont dans `DRAINING` cet état.

> **Note**
>
> Vous ne pouvez pas spécifier de `maximumPercent` valeur personnalisée pour un service qui utilise le type bleu/vert (`CODE_DEPLOY`) ou le type de `EXTERNAL` déploiement et dont les tâches utilisent le type de EC2 lancement.

Si un service utilise les types de déploiement bleu/vert (`CODE_DEPLOY`) ou `EXTERNAL` et exécute des tâches employant le type de lancement Fargate, la valeur de pourcentage minimum d'instances saines n'est pas utilisée, même si elle est renvoyée pour décrire votre service.

Contrôleur de déploiement

`deploymentController`

Type : objet

Obligatoire : non

Le contrôleur de déploiement à utiliser pour le service. Si aucun contrôleur de déploiement n'est spécifié, le contrôleur `ECS` est utilisé. Pour de plus amples informations, veuillez consulter Services Amazon ECS.

`type`

Type : String

Valeurs valides : `ECS` | `CODE_DEPLOY` | `EXTERNAL`

Obligatoire : oui

Type de contrôleur de déploiement à utiliser. Il existe trois types de contrôleurs de déploiement disponibles :

`ECS`

Le type de déploiement de mise à jour propagée (`ECS`) implique que le planificateur de service remplace la version de conteneur actuellement en cours d'exécution par la version la plus récente. Le nombre de conteneurs ajoutés ou supprimés par Amazon ECS du service lors de la propagation des mises à jour est contrôlé en ajustant le nombre minimum et maximum de tâches saines lors d'un déploiement de service autorisé, tel que spécifié dans le deploymentConfiguration.

`CODE_DEPLOY`

Le type de déploiement blue/green (`CODE_DEPLOY`) utilise le modèle de blue/green déploiement optimisé par CodeDeploy, qui vous permet de vérifier un nouveau déploiement d'un service avant de lui envoyer du trafic de production.

`EXTERNAL`

Utilisez le type de déploiement externe lorsque vous souhaitez utiliser n'importe quel contrôleur de déploiement tiers pour contrôler complètement le processus de déploiement d'un service Amazon ECS.

Placement des tâches

`placementConstraints`

Type : tableau d'objets

Obligatoire : non

Tableau d'objets de contraintes de placement à utiliser pour les tâches de votre service. Vous pouvez spécifier un maximum de 10 contraintes par tâche. Cette limite inclut les contraintes de la définition de tâche et celles spécifiées lors de l'exécution. Si vous utilisez le type de lancement Fargate, les contraintes de placement des tâches ne sont pas prises en charge.

`type`

Type : chaîne

Obligatoire : non

Type de contrainte. Utilisez `distinctInstance` pour vous assurer que chaque tâche d'un groupe particulier s'exécute sur une instance de conteneur différente. Utilisez `memberOf` pour limiter la sélection à un groupe de candidats valides. La valeur `distinctInstance` n'est pas prise en charge dans les définitions de tâche.

`expression`

Type : chaîne

Obligatoire : non

Expression de langage de requête de cluster à appliquer à la contrainte. Vous ne pouvez pas spécifier d'expression si le type de contrainte est `distinctInstance`. Pour de plus amples informations, veuillez consulter Création d'expressions pour définir des instances de conteneur pour les tâches Amazon ECS.

`placementStrategy`

Type : tableau d'objets

Obligatoire : non

Objets de stratégie de placement à utiliser pour les tâches de votre service. Vous pouvez spécifier jusqu'à quatre règles de stratégie par service.

`type`

Type : String

Valeurs valides : `random` | `spread` | `binpack`

Obligatoire : non

Type de stratégie de placement. La stratégie de placement `random` place les tâches de façon aléatoire sur les candidats disponibles. La stratégie de placement `spread` répartit le placement sur les candidats disponibles de façon uniforme en fonction du paramètre du `field`. La stratégie `binpack` place les tâches sur les candidats disponibles qui ont la quantité disponible la moins élevée de la ressource spécifiée avec le paramètre `field`. Par exemple, si vous utilisez un BinPack sur la mémoire, une tâche est placée sur l'instance avec la quantité la moins élevée de mémoire restante, mais tout de même suffisante pour exécuter la tâche.

`field`

Type : chaîne

Obligatoire : non

Champ sur lequel appliquer la stratégie de placement. Pour la stratégie de placement `spread`, les valeurs valides sont `instanceId` (ou `host`, qui produit le même effet), ou toute plateforme ou tout attribut personnalisé appliqué à une instance de conteneur, comme `attribute:ecs.availability-zone`. Pour la stratégie de placement `binpack`, les valeurs valides sont `cpu` et `memory`. Pour la stratégie de placement `random`, ce champ n'est pas utilisé.

Étiquettes

`tags`

Type : tableau d'objets

Obligatoire : non

Métadonnées que vous appliquez au service pour faciliter le classement et l'organisation. Chaque étiquette est constituée d'une clé et d'une valeur facultative que vous définissez. Lorsqu'un service est supprimé, les étiquettes sont également supprimées. Un maximum de 50 balises peut être appliqué au service. Pour de plus amples informations, veuillez consulter Balisage des ressources Amazon ECS.

`key`

Type : String

Contraintes de longueur : Longueur minimum de 1. Longueur maximale de 128.

Obligatoire : non

Partie d'une paire clé-valeur qui constitue une étiquette. Une clé est une étiquette générale qui fait office de catégorie pour les valeurs d'étiquette plus spécifiques.

`value`

Type : String

Contraintes de longueur : Longueur minimum de 0. Longueur maximale de 256.

Obligatoire : non

Partie facultative d'une paire clé-valeur qui constitue une étiquette. Une valeur agit comme un descripteur au sein d'une catégorie d'étiquette (clé).

`enableECSManagedTags`

Type : booléen

Valeurs valides : `true` | `false`

Obligatoire : non

Spécifie si utiliser les identifications gérées par Amazon ECS pour les tâches dans le service. La valeur par défaut est `false` si aucune valeur n'est spécifiée. Pour de plus amples informations, veuillez consulter Utiliser des tags pour la facturation.

`propagateTags`

Type : String

Valeurs valides : `TASK_DEFINITION` | `SERVICE`

Obligatoire : non

Indique s'il faut copier les étiquettes de la définition de tâche ou du service dans les tâches du service. Si aucune valeur n'est spécifiée, les étiquettes ne sont pas copiées. Les étiquettes ne peuvent être copiées dans les tâches du service que lors de la création du service. Pour ajouter des balises à une tâche après la création du service ou de la tâche, utilisez l'action d'API `TagResource`.

Configuration réseau

`networkConfiguration`

Type : objet

Obligatoire : non

Configuration réseau du service. Ce paramètre est obligatoire pour les définitions de tâches qui utilisent le mode réseau `awsvpc` afin de recevoir leur propre interface réseau Elastic. Il n'est pas pris en charge avec les autres modes réseau. Si vous utilisez le type de lancement Fargate, le mode réseau `awsvpc` est requis. Pour plus d'informations sur la mise en réseau pour le type de EC2 lancement Amazon, consultezOptions de mise en réseau des tâches Amazon ECS pour le type de EC2 lancement. Pour plus d'informations sur la mise en réseau pour le type de lancement Fargate, consultez les options de mise en réseau des tâches Amazon ECS pour le type de lancement Fargate.

`awsvpcConfiguration`

Type : objet

Obligatoire : non

Objet représentant les sous-réseaux et les groupes de sécurité pour une tâche ou un service.

`subnets`

Type : tableau de chaînes

Obligatoire : oui

Sous-réseaux associés à la tâche ou au service. Il existe une limite de 16 sous-réseaux pouvant être spécifiés selon `awsvpcConfiguration`.

`securityGroups`

Type : tableau de chaînes

Obligatoire : non

Groupes de sécurité associés à la tâche ou au service. Si vous ne spécifiez pas de groupe de sécurité, c'est le groupe de sécurité par défaut du VPC qui est utilisé. Il existe une limite de cinq groupes de sécurité pouvant être spécifiés selon `awsvpcConfiguration`.

`assignPublicIP`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Si l'interface réseau Elastic de la tâche reçoit une adresse IP publique ou non. Si aucune valeur n'est spécifiée, la valeur par défaut de `DISABLED` est utilisée.

`healthCheckGracePeriodSeconds`

Type : entier

Obligatoire : non

Durée, en secondes, pendant laquelle le planificateur de services Amazon ECS ignore les vérifications incorrectes d'Elastic Load Balancing, de VPC Lattice et de l'état des conteneurs après le démarrage d'une tâche. Si vous ne spécifiez aucune valeur de période de grâce pour le bilan de santé, la valeur par défaut de 0 est utilisée. Si vous n'utilisez aucun des bilans de santé, cela signifie qu'il `healthCheckGracePeriodSeconds` n'est pas utilisé.

Si les tâches de votre service mettent un certain temps à démarrer et à répondre, vous pouvez définir un délai de grâce pour le bilan de santé d'une durée maximale de 2 147 483 647 secondes (environ 69 ans). Pendant ce temps, le planificateur de service Amazon ECS ignore l'état du bilan de santé. La période de grâce peut empêcher le planificateur de service de marquer les tâches comme étant non saines et de les arrêter avant qu'elles aient le temps de s'exécuter.

`loadBalancers`

Type : tableau d'objets

Obligatoire : non

Un objet d'équilibreur de charge qui représente l'équilibreur de charge à utiliser avec votre service. Pour les services qui utilisent un équilibreur de charge Application Load Balancer ou un Network Load Balancer, il existe une limite de cinq groupes cibles que vous pouvez attacher à un service.

Une fois que vous avez créé un service, la configuration de l'équilibreur de charge ne peut pas être modifiée à partir de la AWS Management Console. Vous pouvez utiliser le AWS Copilot AWS CLI ou le SDK pour modifier la configuration de l'équilibreur de charge uniquement pour le contrôleur de déploiement `ECS` évolutif, et non AWS CodeDeploy pour le bleu/vert ou externe. AWS CloudFormation Lorsque vous ajoutez, mettez à jour ou supprimez une configuration d'équilibreur de charge, Amazon ECS lance un nouveau déploiement avec la configuration mise à jour d'Elastic Load Balancing. Cela entraîne l'enregistrement et le désenregistrement des tâches auprès des équilibreurs de charge. Nous vous recommandons de vérifier cela dans un environnement de test avant de mettre à jour la configuration d'Elastic Load Balancing. Pour plus d'informations sur la manière de modifier la configuration, consultez le UpdateServicemanuel Amazon Elastic Container Service API Reference.

Pour les équilibreurs de charge Application Load Balancer et les Network Load Balancer, cet objet doit contenir l'ARN du groupe cible de l'équilibreur de charge, le nom du conteneur (tel qu'il apparaît dans une définition de conteneur) et le port du conteneur pour permettre l'accès à partir de l'équilibreur de charge. Lorsqu'une tâche de ce service est placée sur une instance de conteneur, la combinaison d'instance et de port de conteneur est enregistrée comme cible dans le groupe cible spécifié.

`targetGroupArn`

Type : chaîne

Obligatoire : non

Amazon Resource Name (ARN) complet du groupe cible Elastic Load Balancing associés à un service.

L'ARN d'un groupe cible n'est spécifié que si vous utilisez un Application Load Balancer ou un Network Load Balancer.

`loadBalancerName`

Type : chaîne

Obligatoire : non

Nom de l'équilibreur de charge à associer au service.

Si vous utilisez un Application Load Balancer ou un Network Load Balancer, omettez le paramètre de nom de l'équilibreur de charge.

`containerName`

Type : chaîne

Obligatoire : non

Nom du conteneur (tel qu'il apparaît dans une définition de conteneur) à associer à l'équilibreur de charge.

`containerPort`

Type : entier

Obligatoire : non

Port du conteneur à associer à l'équilibreur de charge. Ce port doit correspondre à un élément `containerPort` dans la définition de tâche utilisée par les tâches du service. Pour les tâches utilisant le type de EC2 lancement, l'instance de conteneur doit autoriser le trafic entrant sur le mappage `hostPort` des ports.

`role`

Type : chaîne

Obligatoire : non

Nom abrégé ou ARN complet du rôle IAM qui permet à Amazon ECS d'adresser des appels à votre équilibreur de charge en votre nom. Ce paramètre est uniquement autorisé si vous utilisez un équilibreur de charge avec un seul groupe cible pour votre service, et si votre définition de tâche n'utilise pas le mode réseau `awsvpc`. Si vous spécifiez le paramètre `role`, vous devez également spécifier un objet d'équilibreur de charge avec le paramètre `loadBalancers`.

Si le rôle spécifié possède un chemin d'accès autre que `/`, vous devez spécifier l'ARN de rôle complet (solution recommandée) ou ajouter ce chemin d'accès comme préfixe du nom de rôle. Par exemple, si un rôle nommé `bar` a le chemin d'accès `/foo/`, vous devez spécifier `/foo/bar` comme nom du rôle. Pour plus d'informations, consultez Noms conviviaux et chemins dans le Guide de l'utilisateur IAM.

> **Important**
>
> Si votre compte a déjà créé le rôle lié à un service Amazon ECS service, ce rôle est utilisé par défaut pour votre service, sauf si vous spécifiez un rôle ici. Le rôle lié à un service est

obligatoire si votre définition de tâche utilise le mode réseau awsvpc, auquel cas vous ne devez pas spécifier de rôle ici. Pour de plus amples informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS.

`serviceConnectConfiguration`

Type : objet

Obligatoire : non

Cette configuration permet à ce service de découvrir des services et de s'y connecter, et, inversement, elle permet à d'autres services au sein d'un espace de noms de découvrir ce service et de s'y connecter.

Pour de plus amples informations, veuillez consulter Utilisez Service Connect pour connecter les services Amazon ECS avec des noms abrégés.

`enabled`

Type : booléen

Obligatoire : oui

Spécifie s'il faut utiliser Service Connect avec ce service.

`namespace`

Type : chaîne

Obligatoire : non

Nom abrégé ou Amazon Resource Name (ARN) complet de l'espace de AWS Cloud Map noms à utiliser avec Service Connect. L'espace de noms doit se trouver dans la même Région AWS que le service Amazon ECS et le cluster. Le type d'espace de noms n'a aucune incidence sur Service Connect. Pour plus d'informations à ce sujet AWS Cloud Map, consultez la section Travailler avec les services dans le guide du AWS Cloud Map développeur.

`services`

Type : tableau d'objets

Obligatoire : non

Tableau d'objets de service Service Connect. Il s'agit de noms et d'alias (également appelés points de terminaison) utilisés par d'autres services Amazon ECS pour se connecter à ce service.

Ce champ n'est pas obligatoire pour un service Amazon ECS « client » membre d'un espace de noms uniquement pour se connecter à d'autres services au sein de cet espace de noms. Par exemple, une application frontale qui accepte les demandes entrantes provenant d'un équilibreur de charge attaché au service ou par d'autres moyens.

Un objet sélectionne un port dans la définition de la tâche, attribue un nom au AWS Cloud Map service et un ensemble d'alias (également appelés points de terminaison) et de ports permettant aux applications clientes de faire référence à ce service.

`portName`

Type : String

Obligatoire : oui

Le `portName` doit correspondre au `name` de l'un des `portMappings` de tous les conteneurs dans la définition de tâche de ce service Amazon ECS.

`discoveryName`

Type : chaîne

Obligatoire : non

`discoveryName`Il s'agit du nom du nouveau AWS Cloud Map service créé par Amazon ECS pour ce service Amazon ECS. Ce nom doit être unique dans l'espace de noms AWS Cloud Map .

Si ce champ n'est pas spécifié, `portName` est utilisé.

`clientAliases`

Type : tableau d'objets

Obligatoire : non

Liste des alias clients pour ce service Service Connect. Vous les utilisez pour attribuer des noms qui peuvent être utilisés par les applications clientes. Le nombre maximum d'alias clients dont vous pouvez disposer dans cette est de 1.

Chaque alias (« point de terminaison ») est un nom DNS et un numéro de port que les autres services Amazon ECS (« clients ») peuvent utiliser pour se connecter à ce service.

Chaque nom et chaque combinaison de port doivent être uniques dans l'espace de noms.

Ces noms sont configurés dans chaque tâche du service client, et non dans AWS Cloud Map. Les demandes DNS visant à résoudre ces noms ne quittent pas la tâche et ne sont pas prises en compte dans le quota de demandes DNS par seconde par l'interface réseau Elastic.

`port`

Type : entier

Obligatoire : oui

Numéro de port d'écoute du proxy Service Connect. Ce port est disponible dans toutes les tâches du même espace de noms.

Pour éviter de modifier vos applications dans les services clients Amazon ECS, attribuez-lui le même port que celui que l'application client utilise par défaut.

`dnsName`

Type : chaîne

Obligatoire : non

Le `dnsName` est le nom que vous utilisez dans les applications des tâches client pour vous connecter à ce service. Le nom doit être une étiquette DNS valide.

Si ce champ n'est pas spécifié, la valeur par défaut est `discoveryName.namespace`. Si le `discoveryName` n'est pas spécifié, le `portName` issu de la définition de la tâche est utilisé.

Pour éviter de modifier vos applications dans les services clients Amazon ECS, attribuez-lui le même nom que celui que l'application client utilise par défaut. Par exemple, quelques noms courants sont `database`, db ou le nom en minuscules d'une base de données, comme `mysql` ou `redis`.

`ingressPortOverride`

Type : entier

Obligatoire : non

(Facultatif) Numéro de port que le proxy Service Connect utilise pour écouter.

Utilisez la valeur de ce champ pour contourner le proxy pour le trafic sur le numéro de port spécifié dans le `portMapping` nommé de la définition de tâche de cette application, puis utilisez-la dans les groupes de sécurité de votre Amazon VPC pour autoriser le trafic vers le proxy de ce service Amazon ECS.

En mode `awsvpc`, la valeur par défaut est le numéro du port du conteneur spécifié dans le `portMapping` nommé dans la définition de tâche de cette application. En mode `bridge`, la valeur par défaut correspond au port éphémère du proxy Service Connect.

`logConfiguration`

Type : LogConfigurationObjet

Obligatoire : non

Cela définit l'endroit où les journaux du proxy Service Connect sont publiés. Utilisez les journaux pour le débogage lors d'événements inattendus. Cette configuration définit le paramètre `logConfiguration` dans le conteneur de proxy Service Connect pour chaque tâche de ce service Amazon ECS. Le conteneur de proxy n'est pas spécifié dans la définition de tâche.

Nous vous recommandons d'utiliser la même configuration de journal que les conteneurs d'applications de la définition de tâche pour ce service Amazon ECS. Car FireLens il s'agit de la configuration du journal du conteneur d'applications. Ce n'est pas le conteneur FireLens log router qui utilise l'image du `fluentd` conteneur `fluent-bit` or.

`serviceRegistries`

Type : tableau d'objets

Obligatoire : non

Détails de la configuration de la découverte de service pour votre service. Pour de plus amples informations, veuillez consulter Utilisez la découverte des services pour connecter les services Amazon ECS aux noms DNS.

`registryArn`

Type : chaîne

Obligatoire : non

Amazon Resource Name (ARN) du registre de service. Le registre des services actuellement pris en charge est AWS Cloud Map. Pour plus d'informations, consultez Utilisation des services dans le Guide du développeur AWS Cloud Map .

`port`

Type : entier

Obligatoire : non

Valeur de port utilisée si votre service de découverte de service a spécifié un registre SRV. Ce champ est obligatoire si le mode réseau `awsvpc` et les registres SRV sont utilisés.

`containerName`

Type : chaîne

Obligatoire : non

La valeur du nom du conteneur, déjà spécifiée dans la définition de tâche, à utiliser pour votre service de découverte de service. Cette valeur est spécifiée dans la définition de tâche. Si la définition de tâche que votre tâche de service spécifie utilise le mode réseau `bridge` ou `host`, vous devez spécifier une combinaison `containerName` et `containerPort` à partir de la définition de tâche. Si la définition de tâche spécifiée par votre tâche de service utilise le mode réseau `awsvpc` et qu'un registre DNS de type SRV est utilisé, vous devez spécifier une combinaison `containerName` et `containerPort` ou une valeur `port`, mais pas les deux.

`containerPort`

Type : entier

Obligatoire : non

La valeur du port à utiliser pour votre service de découverte de service. Cette valeur est spécifiée dans la définition de tâche. Si la définition de tâche spécifiée par votre tâche de service utilise le mode réseau `bridge` ou `host`, vous devez spécifier une combinaison `containerName` et `containerPort` à partir de la définition de tâche. Si la définition de tâche spécifiée par votre tâche de service utilise le mode réseau `awsvpc` et qu'un registre DNS de type SRV est utilisé, vous devez spécifier une combinaison `containerName` et `containerPort` ou une valeur `port`, mais pas les deux.

Jeton client

`clientToken`

Type : chaîne

Obligatoire : non

L'identifiant unique, sensible à la casse, que vous devez fournir afin de garantir l'idempotence de la demande. Il peut comporter jusqu'à 32 ASCII caractères.

Rééquilibrage des zones de disponibilité

`availabilityZoneRebalancing`

Type : chaîne

Obligatoire : non

Indique si le service utilise le rééquilibrage des zones de disponibilité. Les valeurs valides sont et. `ENABLED` `DISABLED` Pour plus d'informations sur le rééquilibrage des zones de disponibilité, consultezÉquilibrer un service Amazon ECS entre les zones de disponibilité.

Configurations de volume

`volumeConfigurations`

Type : objet

Obligatoire : non

Configuration qui sera utilisée pour créer des volumes pour les tâches gérées par le service. Seuls les volumes marqués comme tels `configuredAtLaunch` dans la définition de tâche peuvent être configurés à l'aide de cet objet.

Cet objet est requis pour associer des volumes Amazon EBS à des tâches gérées par un service. Pour de plus amples informations, veuillez consulter Utiliser des volumes Amazon EBS avec Amazon ECS.

`name`

Type : String

Obligatoire : oui

Nom d'un volume configuré lors de la création ou de la mise à jour d'un service. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits de soulignement () et _ tirets () sont autorisés. - Cette valeur doit correspondre au nom du volume spécifié dans la définition de la tâche.

`managedEBSVolume`

Type : objet

Obligatoire : non

Configuration de volume utilisée pour créer des volumes Amazon EBS attachés à des tâches gérées par un service lors de sa création ou de sa mise à jour. Un volume est joint par tâche.

`encrypted`

Type : booléen

Obligatoire : non

Valeurs valides : `true|false`

Spécifie s'il faut chiffrer chaque volume Amazon EBS créé. Si vous avez activé le chiffrement Amazon EBS par défaut pour une personne en particulier Région AWS , Compte AWS mais que vous avez défini ce paramètre sur`false`, ce paramètre sera remplacé et les volumes seront chiffrés avec la clé KMS spécifiée pour le chiffrement par défaut. Pour plus d'informations sur le chiffrement Amazon EBS par défaut, consultez la section Activer le chiffrement Amazon EBS par défaut dans le guide de l'utilisateur Amazon EBS. Pour plus d'informations sur le chiffrement des volumes Amazon EBS associés aux tâches Amazon ECS, consultez. Chiffrez les données stockées dans les volumes Amazon EBS associés aux tâches Amazon ECS

`kmsKeyId`

Type : chaîne

Obligatoire : non

Identifiant de la clé AWS Key Management Service (AWS KMS) à utiliser pour le chiffrement Amazon EBS. Si `kmsKeyId` est spécifié, l'état chiffré doit être `true`.

La clé spécifiée à l'aide de ce paramètre remplace la clé par défaut d'Amazon EBS ou toute clé KMS au niveau du cluster pour le chiffrement du stockage géré Amazon ECS que vous avez éventuellement spécifiée. Pour de plus amples informations, veuillez consulter Chiffrez les données stockées dans les volumes Amazon EBS associés aux tâches Amazon ECS.

Vous pouvez spécifier la clé KMS en utilisant l'une des méthodes suivantes :

- ID de clé — Par exemple,`1234abcd-12ab-34cd-56ef-1234567890ab`.
- Alias clé — Par exemple,`alias/ExampleAlias`.
- ARN clé — Par exemple,`arn:aws:kms:us-east-1:012345678910:key/1234abcd-12ab-34cd-56ef-1234567890ab`.
- Alias ARN — Par exemple,`arn:aws:kms:us-east-1:012345678910:alias/ExampleAlias`.

> **Important**
>
> AWS authentifie la clé KMS de manière asynchrone. Par conséquent, si vous spécifiez un ID, un alias ou un ARN non valide, l'action peut sembler réussie, mais elle finit par échouer. Pour plus d'informations, consultez Résolution des problèmes de connexion aux volumes Amazon EBS.

`volumeType`

Type : chaîne

Obligatoire : non

Valeurs valides : `gp2` | `gp3` | `io1` | `io2` | `sc1` | `st1` | `standard`

Type de volume Amazon EBS. Pour plus d'informations sur les types de volumes, consultez les types de volumes Amazon EBS dans le guide de l'utilisateur Amazon EBS. Le type de volume par défaut est `gp3`.

> **Note**
>
> Le type de `standard` volume n'est pas pris en charge pour les tâches Fargate.

`sizeInGiB`

Type : entier

Obligatoire : non

Plage valide : nombres entiers compris entre 1 et 16 384

Taille du volume EBS en gibioctets (GiB). Si vous ne fournissez pas d'ID de capture pour configurer un volume en vue de le joindre, vous devez fournir une valeur de taille. Si vous configurez un volume pour la pièce jointe à l'aide d'un instantané, la valeur par défaut est la taille de l'instantané. Vous pouvez ensuite spécifier une taille supérieure ou égale à la taille de l'instantané.

Pour les types de `gp3` volume `gp2` et, la plage valide est comprise entre 1 et 16 384.

Pour les types de `io2` volume `io1` et, la plage valide est comprise entre 4 et 16 384.

Pour les types de `sc1` volume `st1` et, la plage valide est comprise entre 125 et 16 384.

Pour le type de `standard` volume, la plage valide est comprise entre 1 et 1 024.

`snapshotId`

Type : chaîne

Obligatoire : non

L'ID de l'instantané d'un volume Amazon EBS existant qu'Amazon ECS utilise pour créer de nouveaux volumes à joindre. Vous devez spécifier `snapshotId` ou `sizeInGiB`.

`volumeInitializationRate`

Type : entier

Obligatoire : non

Débit, en Mbits/s, auquel les données sont extraites d'un instantané d'un volume Amazon EBS existant afin de créer de nouveaux volumes à joindre. Cette propriété ne peut être spécifiée que si vous spécifiez un`snapshotId`. Pour plus d'informations sur ce taux d'initialisation de volume, y compris la gamme de taux pris en charge pour l'initialisation, consultez Initialiser les volumes Amazon EBS dans le guide de l'utilisateur Amazon EBS.

`iops`

Type : entier

Obligatoire : non

Nombre d' I/O opérations par seconde (IOPS). Pour les volumes `gp3`, `io1` et `io2`, cela représente le nombre d'IOPS provisionnés pour le volume. Pour les `gp2` volumes, cette valeur représente les performances de base du volume et le taux auquel le volume accumule des I/O crédits en cas d'éclatement. Ce paramètre est requis pour les volumes `io1` et `io2`. Ce paramètre n'est pas pris en charge pour les `standard` volumes `gp2` `st1sc1`,, ou.

Pour les `gp3` volumes, la plage de valeurs valide est comprise entre 3 000 et 16 000.

Pour les `io1` volumes, la plage de valeurs valide est comprise entre 100 et 64 000.

Pour les `io2` volumes, la plage de valeurs valide est comprise entre 100 et 64 000.

`throughput`

Type : entier

Obligatoire : non

Débit à provisionner pour les volumes attachés aux tâches gérées par un service.

> **Important**
>
> Ce paramètre n'est pris en charge que pour les `gp3` volumes.

`roleArn`

Type : String

Obligatoire : oui

L'Amazon Resource ARN (ARN) du rôle d'infrastructure AWS Identity and Access Management (IAM) qui fournit les autorisations Amazon ECS pour gérer les ressources Amazon EBS pour vos tâches. Pour de plus amples informations, veuillez consulter Rôle IAM dans l'infrastructure Amazon ECS.

`tagSpecifications`

Type : objet

Obligatoire : non

Spécification des balises à appliquer à chaque volume Amazon EBS.

`resourceType`

Type : String

Obligatoire : oui

Valeurs valides : `volume`

Le type de ressource à baliser à la création.

`tags`

Type : tableau d'objets

Obligatoire : non

Les métadonnées que vous appliquez aux volumes pour vous aider à les classer et à les organiser. Chaque balise est composée d'une clé et d'une valeur facultative, que vous définissez toutes deux. `AmazonECSCreatedet` `AmazonECSManaged` sont des balises réservées ajoutées par Amazon ECS en votre nom. Vous pouvez donc spécifier vous-même un maximum de 48 balises. Lorsqu'un volume est supprimé, les balises sont également supprimées. Pour de plus amples informations, veuillez consulter Balisage des ressources Amazon ECS.

`key`

Type : String

Contraintes de longueur : Longueur minimum de 1. Longueur maximale de 128.

Obligatoire : non

Partie d'une paire clé-valeur constituant une balise. Une clé est une étiquette générale qui fait office de catégorie pour les valeurs d'étiquette plus spécifiques.

`value`

Type : String

Contraintes de longueur : Longueur minimum de 0. Longueur maximale de 256.

Obligatoire : non

Partie facultative d'une paire clé-valeur qui constitue une balise. Une valeur agit comme un descripteur au sein d'une catégorie d'étiquette (clé).

`propagateTags`

Type : String

Valeurs valides : `TASK_DEFINITION` | `SERVICE` | `NONE`

Obligatoire : non

Spécifie s'il faut copier les balises de la définition de tâche ou du service vers un volume. Si cette valeur `NONE` est spécifiée ou si aucune valeur n'est spécifiée, les balises ne sont pas copiées.

`fileSystemType`

Type : chaîne

Obligatoire : non

Valeurs valides : `xfs` | `ext3` | `ext4` | `NTFS`

Type de système de fichiers sur un volume. Le type de système de fichiers du volume détermine la manière dont les données sont stockées et récupérées dans le volume. Pour les volumes créés à partir d'un instantané, vous devez spécifier le même type de système de fichiers que celui utilisé par le volume lors de la création de l'instantané. Si le type de système de fichiers ne correspond pas, la tâche ne démarrera pas.

Les valeurs valides pour Linux sont `xfs` ext3`, and ext4`. La valeur par défaut pour les volumes attachés à des tâches Linux est`XFS`.

Les valeurs valides pour Windows sont`NTFS`. La valeur par défaut pour les volumes attachés à des tâches Windows est`NTFS`.

Modèle de définition de service

Voici une illustration de la représentation JSON d'une définition de service Amazon ECS service.

Type de EC2 lancement Amazon

```
{
    "cluster": "",
    "serviceName": "",
    "taskDefinition": "",
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "loadBalancerName": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "desiredCount": 0,
    "clientToken": "",
    "launchType": "EC2",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "role": "",
    "deploymentConfiguration": {
        "deploymentCircuitBreaker": {
            "enable": true,
            "rollback": true
        },
        "maximumPercent": 0,
        "minimumHealthyPercent": 0,
        "alarms": {
            "alarmNames": [
                ""
```

```
            ],
            "enable": true,
            "rollback": true
        }
    },
    "placementConstraints": [
        {
            "type": "distinctInstance",
            "expression": ""
        }
    ],
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "healthCheckGracePeriodSeconds": 0,
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION",
    "enableExecuteCommand": true,
    "availabilityZoneRebalancing": "ENABLED",
    "serviceConnectConfiguration": {
```

```
        "enabled": true,
        "namespace": "",
        "services": [
            {
                "portName": "",
                "discoveryName": "",
                "clientAliases": [
                    {
                        "port": 0,
                        "dnsName": ""
                    }
                ],
                "ingressPortOverride": 0
            }
        ],
        "logConfiguration": {
            "logDriver": "journald",
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ]
        }
    },
    "volumeConfigurations": [
        {
            "name": "",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "",
                "volumeType": "",
                "sizeInGiB": 0,
                "snapshotId": "",
                "volumeInitializationRate": 0,
                "iops": 0,
                "throughput": 0,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
```

```
                            {
                                "key": "",
                                "value": ""
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "",
                "filesystemType": ""
            }
        }
    ]
}
```

Type de lancement Fargate

```
{
    "cluster": "",
    "serviceName": "",
    "taskDefinition": "",
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "loadBalancerName": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "desiredCount": 0,
    "clientToken": "",
    "launchType": "FARGATE",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
```

```
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "platformFamily": "",
    "role": "",
    "deploymentConfiguration": {
        "deploymentCircuitBreaker": {
            "enable": true,
            "rollback": true
        },
        "maximumPercent": 0,
        "minimumHealthyPercent": 0,
        "alarms": {
            "alarmNames": [
                ""
            ],
            "enable": true,
            "rollback": true
        }
    },
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "healthCheckGracePeriodSeconds": 0,
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
```

```
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION",
    "enableExecuteCommand": true,
    "availabilityZoneRebalancing": "ENABLED",
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "",
        "services": [
            {
                "portName": "",
                "discoveryName": "",
                "clientAliases": [
                    {
                        "port": 0,
                        "dnsName": ""
                    }
                ],
                "ingressPortOverride": 0
            }
        ],
        "logConfiguration": {
            "logDriver": "journald",
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ]
        }
    },
    "volumeConfigurations": [
        {
            "name": "",
            "managedEBSVolume": {
                "encrypted": true,
```

```
                "kmsKeyId": "",
                "volumeType": "",
                "sizeInGiB": 0,
                "snapshotId": "",
                "volumeInitializationRate": 0,
                "iops": 0,
                "throughput": 0,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "",
                                "value": ""
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "",
                "filesystemType": ""
            }
        }
    ]
}
```

Vous pouvez créer ce modèle de définition de service à l'aide de la commande AWS CLI suivante.

```
aws ecs create-service --generate-cli-skeleton
```

Balisage des ressources Amazon ECS

Pour vous aider à gérer vos ressources Amazon ECS, vous pouvez éventuellement affecter vos propres métadonnées à chaque ressource à l'aide de balises. Chaque balise est constituée d'une clé et d'une valeur facultative.

Vous pouvez utiliser des étiquettes pour classer vos ressources Amazon ECS de différentes manières, par exemple, par objectif, par propriétaire ou par environnement. Cela est utile lorsque vous avez de nombreuses ressources du même type. Vous pouvez identifier rapidement une ressource spécifique en fonction des étiquettes que vous lui avez attribuées. Par exemple, vous pouvez définir pour les instances de votre compte Amazon ECS un ensemble d'étiquettes. Cela vous aide à suivre le propriétaire et le niveau de la pile de chaque instance.

Vous pouvez utiliser des balises pour vos Rapports d'utilisation et de coût. Vous pouvez utiliser ces rapports pour analyser le coût et de l'utilisation de vos ressources Amazon ECS. Pour de plus amples informations, veuillez consulter the section called “Rapports d'utilisation”.

> **Warning**
>
> Il y en a beaucoup APIs qui renvoient les clés des balises et leurs valeurs. Le fait de refuser l'accès à `DescribeTags` ne refuse pas automatiquement l'accès aux balises renvoyées par d'autres APIs. Nous vous recommandons de ne pas inclure de données sensibles dans vos balises.

Nous vous recommandons de concevoir un ensemble de clés d’étiquette répondant à vos besoins pour chaque type de ressource. Vous pouvez utiliser un ensemble de clés de balise cohérent facilite la gestion de vos ressources. Vous pouvez rechercher et filtrer les ressources en fonction des étiquettes que vous ajoutez.

Les balises n'ont pas de signification sémantique pour Amazon ECS et sont interprétées strictement comme des chaînes de caractères. Vous pouvez modifier les clés et valeurs de balise, et vous pouvez retirer des balises d’une ressource à tout moment. Vous pouvez définir la valeur d'une étiquette sur une chaîne vide, mais vous ne pouvez pas définir la valeur d'une étiquette sur null. Si vous ajoutez une balise ayant la même clé qu'une balise existante sur cette ressource, la nouvelle valeur remplace l'ancienne valeur. Lorsque vous supprimez une ressource, toutes les balises associées à cette ressource sont également supprimées.

Si vous utilisez AWS Identity and Access Management (IAM), vous pouvez contrôler quels utilisateurs de votre AWS compte sont autorisés à gérer les tags.

Comment les ressources sont étiquetées

Les tâches, les services, les définitions de tâches Amazon ECS sont balisés de différentes manières :

- Un utilisateur balise manuellement une ressource à l' AWS Management Console aide de l'API Amazon ECS AWS CLI, du ou d'un AWS SDK.
- Un utilisateur crée un service ou exécute une tâche autonome et sélectionne l'option de balises gérées par Amazon ECS.

  Amazon ECS étiquette automatiquement toutes les tâches récemment lancées. Pour de plus amples informations, veuillez consulter the section called “Balises gérées par Amazon ECS”.
- Un utilisateur crée une ressource à l'aide de la console. La console étiquette automatiquement les ressources.

  Ces balises sont renvoyées dans AWS CLI les réponses du AWS SDK et sont affichées dans la console. Vous ne pouvez pas modifier ni supprimer ces balises.

  Pour plus d'informations sur les balises ajoutées, consultez la colonne Balises ajoutées automatiquement par la console dans le tableau des ressources relatives à la prise en charge du balisage pour Amazon ECS.

Si vous spécifiez des balises lorsque vous créez une ressource et que les balises ne peuvent pas être appliquées, Amazon ECS annule le processus de création. Cela permet de s'assurer que les ressources sont créées avec des étiquettes ou qu'elles ne sont pas créées du tout, et qu'aucune ressource ne demeurent sans étiquette à tout moment. En attribuant des étiquettes aux ressources au moment de la création, vous pouvez supprimer la nécessité d'exécuter des scripts d'identification personnalisés après la création de ressources.

Le tableau suivant décrit les ressources Amazon ECS qui prennent en charge les balises.

Ressource	Prend en charge les étiquettes	Prend en charge la propagation des étiquettes	Balises ajoutées automatiquement par la console
Tâches Amazon ECS	Oui	Oui, à partir de la définition de tâche.	Clé : `aws:ecs:clusterName` Value (Valeur) : `cluster-name`
Services Amazon ECS	Oui	Oui, à partir soit de la définition de tâche ou de service pour les tâches dans le service.	Clé : `ecs:service:stackId` Value (Valeur) : `arn:aws:cloudformation:` *`arn`*
Ensembles de tâches Amazon ECS	Oui	Non	N/A
Définitions de tâche Amazon ECS	Oui	Non	Clé : `ecs:taskDefinition:createdFrom` Value (Valeur) : `ecs-console-v2`
Clusters Amazon ECS	Oui	Non	Clé : `aws:cloudformation:logical-id` Value (Valeur) : `ECSCluster` Clé : `aws:cloudformation:stack-id`

Ressource	Prend en charge les étiquettes	Prend en charge la propagation des étiquettes	Balises ajoutées automatiquement par la console
			Value (Valeur) : `arn:aws:cloudformation:` *`arn`* Clé : `aws:cloudformation:stack-name` Value (Valeur) : `ECS-Console-V2-Cluster-` *`EXAMPLE`*
Instances de conteneur Amazon ECS	Oui	Oui, depuis l' EC2 instance Amazon. Pour de plus amples informations, veuillez consulter Ajouter des balises à une instance de conteneur Amazon ECS.	N/A
Instances Amazon ECS externes	Oui	Non	N/A
Fournisseur de capacité Amazon ECS	Oui. Vous ne pouvez pas étiqueter les fournisseurs prédéfinis de capacité `FARGATE` et `FARGATE_SPOT` .	Non	N/A

Balisage des ressources à la création

Les ressources suivantes prennent en charge le balisage lors de la création à l'aide de l'API ou du AWS SDK Amazon ECS : AWS CLI

- Tâches Amazon ECS
- Services Amazon ECS
- Définition de tâche Amazon ECS
- Ensembles de tâches Amazon ECS
- Clusters Amazon ECS
- Instances de conteneur Amazon ECS
- Fournisseurs de capacité Amazon ECS

Amazon ECS a la possibilité d'utiliser l'autorisation de balisage pour la création de ressources. Lorsque le Compte AWS est configuré pour l'autorisation de balisage, les utilisateurs doivent disposer d'autorisations pour les actions qui créent la ressource, telles que`ecsCreateCluster`. Si vous spécifiez des balises dans l'action de création de ressources, AWS effectue une autorisation supplémentaire pour vérifier si les utilisateurs ou les rôles sont autorisés à créer des balises. Par conséquent, vous devez octroyer des autorisations explicites d'utiliser l'action `ecs:TagResource`. Pour de plus amples informations, veuillez consulter the section called “Baliser des ressources pendant la création”. Pour plus d'informations sur la configuration de l'option, consultezthe section called “Autorisation de balisage”.

Restrictions

Les restrictions suivantes s’appliquent aux balises :

- Un maximum de 50 balises peut être associé à une ressource.
- Les clés de balises ne peuvent pas être répétées pour une même ressource. Chaque clé de balise doit être unique et ne peut avoir qu'une seule valeur.
- Les clés peuvent contenir jusqu'à 128 caractères en UTF-8.
- Les valeurs peuvent contenir jusqu'à 256 caractères en UTF-8.
- Si plusieurs Services AWS ressources utilisent votre schéma de balisage, limitez les types de caractères que vous utilisez. Certains services peuvent présenter des restrictions sur les

caractères autorisés. Les caractères généralement autorisés sont les lettres, les chiffres, les espaces et les caractères suivants : **+ - = . _ : / @**.

- Les clés et valeurs de balise sont sensibles à la casse.
- Vous ne pouvez pas utiliser `aws:`, `AWS:` ou n'importe quelle combinaison de majuscules ou minuscules de ce préfixe pour des clés ou des valeurs Ils sont réservés uniquement à AWS l'usage. Vous ne pouvez pas modifier ni supprimer des clés ou valeurs d'étiquette ayant ce préfixe. Les balises comportant ce préfixe ne sont pas prises en compte dans votre tags-per-resource limite.

Balises gérées par Amazon ECS

Lorsque vous utilisez des balises gérées par Amazon ECS, Amazon ECS étiquette automatiquement toutes les tâches récemment lancées et tous les volumes Amazon EBS attachés aux tâches avec les informations du cluster et soit les balises de définition de tâche ajoutées par l'utilisateur, soit les balises de service. Les balises ajoutées sont décrites ci-dessous :

- Tâches autonomes : une balise avec une clé en tant que `aws:ecs:clusterName` et une valeur définies en fonction du nom du cluster. Toutes les balises de définition de tâches qui ont été ajoutées par les utilisateurs. Un volume Amazon EBS associé à une tâche autonome recevra le tag avec une clé `aws:ecs:clusterName` et une valeur définies sur le nom du cluster. Pour plus d'informations sur le balisage des volumes Amazon EBS, consultez la section Marquage des volumes Amazon EBS.
- Tâches faisant partie d'un service : une balise avec une clé en tant que `aws:ecs:clusterName` et une valeur définies en fonction du nom du cluster. Balise avec une clé en tant que `aws:ecs:serviceName` et une valeur définies en fonction du nom du service. Balises de l'une des ressources suivantes.
  - Définitions de tâches : toutes les balises de définition de tâches qui ont été ajoutées par les utilisateurs.
  - Services : toutes les balises de service qui ont été ajoutées par les utilisateurs.

    Un volume Amazon EBS attaché à une tâche faisant partie d'un service recevra une balise avec une clé sous forme `aws:ecs:clusterName` et une valeur définies sur le nom du cluster, et une balise avec une clé en tant que `aws:ecs:serviceName` et une valeur définies sur le nom du service. Pour plus d'informations sur le balisage des volumes Amazon EBS, consultez la section Marquage des volumes Amazon EBS.

Les options suivantes sont requises pour cette fonctionnalité :

- Vous devez vous inscrire au nouveaux formats Amazon Resource Name (ARN) et identifiants de ressource (ID). Pour de plus amples informations, veuillez consulter Amazon Resource Names (ARNs) et IDs.
- Lorsque vous utilisez le APIs pour créer un service ou exécuter une tâche, vous devez définir `enableECSManagedTags` `true` pour `run-task` et`create-service`. Pour plus d'informations, consultez create-service et run-task dans la Référence API d'AWS Command Line Interface .
- Amazon ECS utilise des balises gérées pour déterminer à quel moment certaines fonctionnalités sont activées, par exemple l'autoscaling de cluster. Nous vous recommandons de ne pas modifier manuellement les balises afin qu'Amazon ECS puisse gérer efficacement les fonctionnalités.

Utiliser des tags pour la facturation

AWS fournit un outil de reporting appelé Cost Explorer que vous pouvez utiliser pour analyser le coût et l'utilisation de vos ressources Amazon ECS.

Vous pouvez utiliser Cost Explorer pour afficher les graphiques de votre utilisation et de vos coûts. Vous pouvez afficher les données des 13 derniers mois et prévoir vos dépenses pour les trois prochains mois. Vous pouvez utiliser Cost Explorer pour afficher des schémas de vos dépenses en ressources AWS au fil du temps. Par exemple, vous pouvez l'utiliser pour identifier les zones qui méritent d'être approfondies et connaître les tendances que vous pouvez utiliser pour comprendre vos coûts. Vous pouvez également spécifier des plages de temps pour les données et afficher des données temporelles par jour ou par mois.

Vous pouvez utiliser des balises gérées par Amazon ECS ou des balises ajoutées par l'utilisateur pour votre rapport sur les coûts et l'utilisation. Pour de plus amples informations, veuillez consulter Rapports d'utilisation d'Amazon ECS.

Pour voir le coût de vos ressources combinées, vous pouvez organiser vos informations de facturation en fonction des ressources possédant les mêmes valeurs de clé d‘étiquette. Par exemple, vous pouvez étiqueter plusieurs ressources avec un nom d'application spécifique, puis organiser vos informations de facturation pour afficher le coût total de cette application dans plusieurs services. Pour en savoir plus sur la configuration d'un rapport de répartition des coûts avec des étiquettes, consultez Rapport d'allocation des coûts mensuel dans le guide de l'utilisateur AWS Billing .

En outre, vous pouvez activer les données de répartition des coûts fractionnés pour obtenir des données d'utilisation de la mémoire et du processeur au niveau des tâches dans vos rapports

d'utilisation et de coût. Pour de plus amples informations, veuillez consulter Rapports d'utilisation et de coût au niveau des tâches.

> **Note**
>
> Si vous avez activé la création de rapports, 24 heures peuvent être nécessaires avant que les données du mois en cours puissent être consultées.

Ajouter des balises aux ressources Amazon ECS

Vous pouvez étiqueter des tâches, des services, des définitions de tâches ou des clusters nouveaux ou existants. Pour plus d'informations sur le balisage de vos instances de conteneur, consultezAjouter des balises à une instance de conteneur Amazon ECS.

> **Warning**
>
> N'ajoutez pas de données d'identification personnelle (PII) ou d'autres informations confidentielles ou sensibles dans les étiquettes. Les tags sont accessibles à de nombreux AWS services, y compris la facturation. Les étiquettes ne sont pas destinées à être utilisées pour des données privées ou sensibles.

Vous pouvez utiliser les ressources suivantes pour spécifier des balises lorsque vous créez la ressource.

Tâche	console	AWS CLI	Action d'API
Exécuter une ou plusieurs tâches.	Exécution d'une application en tant que tâche Amazon ECS	run-task	RunTask
Créer un service.	Création d'un déploiement de mises à jour continues Amazon ECS	create-service	CreateService

Tâche	console	AWS CLI	Action d'API
Créer un ensemble de tâches.	Déployez les services Amazon ECS à l'aide d'un contrôleur tiers	create-task-set	CreateTaskSet
Enregistrer une définition de tâche.	the section called “Création d'une définition de tâche à l'aide de la console”	register-task-definition	RegisterTaskDefinition
Créer un cluster.	Création d'un cluster Amazon ECS pour le type de lancement Fargate	create-cluster	CreateCluster
Exécuter une ou plusieurs instances de conteneur.	Lancement d'une instance de conteneur Amazon ECS Linux	run-instances	RunInstances

Ajouter des balises aux ressources existantes (console Amazon ECS)

Vous pouvez ajouter ou supprimer des balises associées à vos clusters, services, tâches et définitions de tâches directement depuis la page de la ressource.

Pour modifier une balise à une ressource individuelle

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans la barre de navigation, sélectionnez le Région AWS à utiliser.
3. Dans le panneau de navigation, sélectionnez un type de ressource (par exemple, Clusters).
4. Sélectionnez la ressource à partir de la liste des ressources et choisissez l'onglet Tags (Balises), puis Manage Tags (Gérer les balises).

5. Configurez vos balises.

   [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :

   - Pour Clé, saisissez le nom de la clé.
   - Pour Valeur, saisissez la valeur de clé.

6. Choisissez Enregistrer.

Ajouter des balises aux ressources existantes (AWS CLI)

Vous pouvez ajouter ou remplacer une ou plusieurs balises à l'aide de l'API AWS CLI ou d'une API.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir de AWS CLI SDKs, et de l'API Amazon ECS via les deux IPv4 et. IPv6 Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

- AWS CLI - tag-ressource
- Action de l'API - TagResource

Ajouter des balises à une instance de conteneur Amazon ECS

Vous pouvez associer des étiquettes à vos instances de conteneur en utilisant l'une des méthodes suivantes :

- Méthode 1 — Lorsque vous créez l'instance de conteneur à l'aide de l' EC2API, de la CLI ou de la console Amazon, spécifiez les balises en transmettant les données utilisateur à l'instance à l'aide du paramètre de configuration de l'agent de conteneur`ECS_CONTAINER_INSTANCE_TAGS`. Cela crée des balises associées à l'instance de conteneur dans Amazon ECS uniquement, elles ne peuvent pas être répertoriées à l'aide de l' EC2 API Amazon. Pour de plus amples informations, veuillez consulter Démarrage des instances de conteneur Linux Amazon ECS pour transmettre des données.

> **Important**
>
> Si vous lancez vos instances de conteneur à l'aide d'un groupe Amazon EC2 Auto Scaling, vous devez utiliser le paramètre de configuration de l'agent ECS_CONTAINER_INSTANCE_TAGS pour ajouter des balises. Cela est dû à la manière dont les balises sont ajoutées aux EC2 instances Amazon lancées à l'aide de groupes Auto Scaling.

Voici un exemple de script de données utilisateur qui associe des étiquettes avec votre instance de conteneur :

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
EOF
```

- Méthode 2 — Lorsque vous créez votre instance de conteneur à l'aide de l' EC2 API, de la CLI ou de la console Amazon, spécifiez d'abord les balises à l'aide du `TagSpecification.N` paramètre. Transmettez ensuite les données utilisateur à l'instance à l'aide du paramètre de configuration de l'agent de conteneur `ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM`. Cela les propage d'Amazon EC2 vers Amazon ECS.

  Voici un exemple de script de données utilisateur qui propage les balises associées à une EC2 instance Amazon et enregistre l'instance auprès d'un cluster nommé`MyCluster`.

  ```
  #!/bin/bash
  cat <<'EOF' >> /etc/ecs/ecs.config
  ECS_CLUSTER=MyCluster
  ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM=ec2_instance
  EOF
  ```

  Pour fournir un accès permettant aux balises d'instance de conteneur de se propager d'Amazon EC2 vers Amazon ECS, ajoutez manuellement les autorisations suivantes en tant que politique intégrée au rôle IAM de l'instance de conteneur Amazon ECS. Pour plus d'informations, consultez Ajout et suppression de politiques IAM.

  - `ec2:DescribeTags`

L'exemple suivant de stratégie ajoute ces autorisations.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
             "ec2:DescribeTags"
            ],
            "Resource": "*"
        }
    ]
}
```

Instances de conteneurs externes

Vous pouvez associer des balises à vos instances de conteneur externes en utilisant l'une des méthodes suivantes.

- Méthode 1 – Avant d'exécuter le script d'installation pour enregistrer votre instance externe auprès de votre cluster, créez ou modifiez le fichier de configuration de l'agent de conteneur Amazon ECS à l'adresse `/etc/ecs/ecs.config` et ajoutez le paramètre de configuration `ECS_CONTAINER_INSTANCE_TAGS` de l'agent de conteneur. Cela permet de créer des balises associées à l'instance externe.

  Voici un exemple de syntaxe.

  ```
  ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
  ```

- Méthode 2 — Une fois que votre instance externe est enregistrée dans votre cluster, vous pouvez utiliser le AWS Management Console pour ajouter des balises. Pour de plus amples informations, veuillez consulter Ajouter des balises aux ressources existantes (console Amazon ECS).

Rapports d'utilisation d'Amazon ECS

AWS fournit un outil de reporting appelé Cost Explorer que vous pouvez utiliser pour analyser le coût et l'utilisation de vos ressources Amazon ECS.

Vous pouvez utiliser Cost Explorer pour afficher les graphiques de votre utilisation et de vos coûts. Vous pouvez afficher les données des 13 derniers mois et prévoir vos dépenses pour les trois prochains mois. Vous pouvez utiliser Cost Explorer pour afficher des schémas de vos dépenses en ressources AWS au fil du temps. Par exemple, vous pouvez l'utiliser pour identifier les zones qui méritent d'être approfondies et connaître les tendances que vous pouvez utiliser pour comprendre vos coûts. Vous pouvez également spécifier des plages de temps pour les données et afficher des données temporelles par jour ou par mois.

Les données de mesure de votre rapport de coût et d'utilisation illustre l'utilisation parmi toutes vos tâches Amazon ECS. Les données de métrique incluent l'utilisation du processeur comme `vCPU-Hours` et l'utilisation de la mémoire comme `GB-Hours` pour chaque tâche exécutée. La présentation de ces données dépend du type de lancement de la tâche.

Pour les tâches utilisant le type de lancement Fargate, la colonne `lineItem/Operation` affiche `FargateTask` et vous permet de connaître le coût associé à chaque tâche.

Pour les tâches qui utilisent le type de EC2 lancement, la `lineItem/Operation` colonne apparaît `ECSTask-EC2` et aucun coût direct n'est associé aux tâches. Les données de mesure indiquées dans le rapport, telles que l'utilisation de la mémoire, représentent le total des ressources réservées par la tâche au cours de la période de facturation que vous spécifiez. Vous pouvez utiliser ces données pour déterminer le coût de votre cluster sous-jacent d' EC2 instances Amazon. Les données de coût et d'utilisation de vos EC2 instances Amazon sont répertoriées séparément dans le EC2 service Amazon.

Vous pouvez également utiliser les balises gérées par Amazon ECS pour identifier le service ou cluster auquel chaque tâche appartient. Pour de plus amples informations, veuillez consulter Utiliser des tags pour la facturation.

> **Important**
>
> Les données de mesure sont uniquement visualisables pour les tâches lancées à compter du 16 novembre 2018. Les tâches lancées avant cette date n'affichent pas les données de mesure.

Voici un exemple des champs que vous pouvez trier par données d'attribution de coût dans Cost Explorer.

- Nom du cluster
- Nom du service
- Étiquettes de ressources
- Type de lancement
- Région AWS
- Type d'utilisation

Pour plus d'informations sur la création d'un rapport sur les AWS coûts et l'utilisation, consultez le rapport sur les AWS coûts et l'utilisation dans le guide de AWS Billing l'utilisateur.

Rapports d'utilisation et de coût au niveau des tâches

AWS Cost Management peut fournir des données d'utilisation du processeur et de la mémoire AWS Cost and Usage Report pour chaque tâche sur Amazon ECS, y compris les tâches sur Fargate et les tâches sur. EC2 Ces données sont appelées données de répartition des coûts fractionnés. Vous pouvez utiliser ces données pour analyser les coûts et l'utilisation des applications. En outre, vous pouvez fractionner et répartir les coûts entre les différentes unités commerciales et équipes à l'aide de balises de répartition des coûts et de catégories de coûts. Pour plus d'informations sur les données de répartition des coûts fractionnés, voir Comprendre les données de répartition des coûts partagés dans le Guide de AWS Cost and Usage Report l'utilisateur.

Vous pouvez activer les données de répartition des coûts fractionnés au niveau des tâches pour le compte dans l' AWS Cost Management Console. Si vous avez un compte de gestion (payeur), vous pouvez choisir d'appliquer cette configuration à chaque compte associé depuis le compte payeur.

Une fois que vous avez configuré les données de répartition des coûts, des colonnes supplémentaires apparaîtront sous l'splitLineItemen-tête du rapport. Pour plus d'informations, voir les détails relatifs aux articles divisés dans le guide de AWS Cost and Usage Report l'utilisateur

Pour les tâches sur EC2, ces données divisent le coût de l' EC2 instance en fonction de l'utilisation des ressources ou des réservations et des ressources restantes de l'instance.

Les conditions requises sont les suivantes :

- Définissez le paramètre de configuration de l'agent `ECS_DISABLE_METRICS` Amazon ECS sur`false`.

  Lorsque ce paramètre est défini`false`, l'agent Amazon ECS envoie des métriques à Amazon CloudWatch. Sous Linux, ce paramètre est défini `false` par défaut et les métriques sont envoyées à CloudWatch. Sous Windows, ce paramètre est défini `true` par défaut. Vous devez donc le modifier `false` CloudWatch pour envoyer les mesures AWS Cost Management à utiliser. Pour en savoir plus sur la configuration de l'agent ECS, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.
- La version minimale de Docker pour des métriques fiables est la version v20.10.13 et les versions ultérieures, qui sont incluses dans l'AMI 20220607 optimisée pour Amazon ECS et les versions plus récentes.

Pour utiliser les données de répartition des coûts fractionnés, vous devez créer un rapport et sélectionner Données de répartition des coûts fractionnés. Pour plus d'informations, consultez la section Création de rapports sur les coûts et l'utilisation dans le guide de AWS Cost and Usage Report l'utilisateur.

AWS Cost Management calcule les données de répartition des coûts en fonction du processeur de la tâche et de l'utilisation de la mémoire. AWS Cost Management peut utiliser le processeur de tâche et la réservation de mémoire au lieu de l'utilisation, si l'utilisation n'est pas disponible. Si vous constatez que le CUR utilise les réservations, vérifiez que vos instances de conteneur répondent aux conditions requises et que les mesures d'utilisation des ressources des tâches apparaissent dans CloudWatch.

Surveillance d'Amazon ECS

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances d'Amazon ECS et de vos AWS solutions. Vous devez collecter des données de surveillance provenant de toutes les parties de votre AWS solution afin de pouvoir corriger plus facilement une défaillance multipoint, le cas échéant. Avant de commencer à surveiller Amazon ECS, créez un plan de surveillance qui inclut les réponses aux questions suivantes :

- Quels sont les objectifs de la surveillance ?
- Quelles sont les ressources à surveiller ?
- À quelle fréquence les ressources doivent-elles être surveillées ?
- Quels outils de surveillance utiliser ?
- Qui exécute les tâches de supervision ?
- Qui doit être informé en cas de problème ?

Les métriques disponibles dépendent du type de lancement des tâches et des services dans vos clusters. Si vous utilisez le type de lancement Fargate pour vos services, les métriques d'utilisation de l'UC et de la mémoire sont fournies pour faciliter leur surveillance. Pour le type de EC2 lancement Amazon, vous possédez et devez surveiller les EC2 instances qui constituent votre infrastructure sous-jacente. Des mesures supplémentaires de réservation et d'utilisation du processeur et de la mémoire sont mises à disposition au niveau du cluster, du service et de la tâche.

L'étape suivante consiste à établir une référence de performances Amazon ECS normales dans votre environnement, en mesurant la performance à divers moments et dans diverses conditions de charge. Lorsque vous surveillez Amazon ECS, conservez les données d'historique de surveillance afin de pouvoir les comparer aux données de performances actuelles, d'identifier les modèles de performances normales et les anomalies de performances, et de concevoir des méthodes pour résoudre les problèmes.

Pour établir une référence, vous devez, au moins, superviser les éléments suivants :

- Métriques de réservation et d'utilisation de l'UC et de la mémoire pour les clusters Amazon ECS
- Métriques d'utilisation de l'UC et de la mémoire pour les services Amazon ECS

  Pour de plus amples informations, veuillez consulter Affichage des métriques Amazon ECS.

Bonnes pratiques pour la surveillance d'Amazon ECS

Utilisez les meilleures pratiques suivantes pour surveiller Amazon ECS.

- Faites de la surveillance une priorité pour éviter les petits problèmes avant qu'ils ne s'aggravent
- Créez un plan de surveillance qui inclut les réponses à la question suivante
  - Quels sont les objectifs de la surveillance ?
  - Quelles sont les ressources à surveiller ?
  - À quelle fréquence les ressources doivent-elles être surveillées ?
  - Quels outils de surveillance utiliser ?
  - Qui exécute les tâches de supervision ?
  - Qui doit être informé en cas de problème ?
- Automatisez la surveillance autant que possible.
- Consultez les fichiers journaux Amazon ECS. Pour de plus amples informations, veuillez consulter Afficher les journaux des agents de conteneurs Amazon ECS.
- Utilisez la surveillance du temps d'exécution pour protéger vos comptes, vos conteneurs, vos charges de travail et les données de votre AWS environnement. Pour de plus amples informations, veuillez consulter Identifiez les comportements non autorisés grâce à la surveillance du temps d'exécution.

Outils de surveillance pour Amazon ECS

AWS fournit différents outils que vous pouvez utiliser pour surveiller Amazon ECS. Vous pouvez configurer certains outils pour qu’ils effectuent la supervision automatiquement, tandis que d’autres nécessitent une intervention manuelle. Nous vous recommandons d’automatiser le plus possible les tâches de supervision.

Outils de surveillance automatique

Vous pouvez utiliser les outils de surveillance automatique pour surveiller Amazon ECS et signaler un problème éventuel :

- CloudWatch Alarmes Amazon : surveillez une seule métrique sur une période que vous spécifiez et effectuez une ou plusieurs actions en fonction de la valeur de la métrique par rapport à un seuil donné sur un certain nombre de périodes. L'action est une notification envoyée à une rubrique

Amazon Simple Notification Service (Amazon SNS) ou à une politique Amazon EC2 Auto Scaling. CloudWatch les alarmes n'appellent pas d'actions simplement parce qu'elles sont dans un état particulier ; l'état doit avoir changé et être maintenu pendant un certain nombre de périodes. Pour de plus amples informations, veuillez consulter Surveillez Amazon ECS à l'aide de CloudWatch.

Pour les services dont les tâches utilisent le type de lancement Fargate, vous pouvez CloudWatch utiliser des alarmes pour augmenter ou réduire les tâches de votre service en CloudWatch fonction de mesures telles que l'utilisation du processeur et de la mémoire. Pour de plus amples informations, veuillez consulter Faites évoluer automatiquement votre service Amazon ECS.

Pour les clusters dont les tâches ou les services utilisent le type de EC2 lancement, vous pouvez utiliser des CloudWatch alarmes pour augmenter ou réduire les instances de conteneur en fonction de CloudWatch métriques, telles que la réservation de mémoire du cluster.

Pour vos instances de conteneur lancées avec l'AMI Amazon Linux optimisée pour Amazon ECS, vous pouvez utiliser CloudWatch Logs pour afficher les différents journaux de vos instances de conteneur en un seul endroit pratique. Vous devez installer l' CloudWatch agent sur vos instances de conteneur. Pour plus d'informations, consultez Télécharger et configurer l' CloudWatch agent à l'aide de la ligne de commande du guide de CloudWatch l'utilisateur Amazon. Vous devez également ajouter la stratégie `ECS-CloudWatchLogs` au rôle `ecsInstanceRole`. Pour de plus amples informations, veuillez consulter Surveillance des autorisations des instances de conteneurs.

- Amazon CloudWatch Logs — Surveillez, stockez et accédez aux fichiers journaux depuis les conteneurs de vos tâches Amazon ECS en spécifiant le pilote de `awslogs` journal dans vos définitions de tâches. Pour de plus amples informations, veuillez consulter Envoyez les journaux Amazon ECS à CloudWatch .

  Vous pouvez également surveiller, stocker et accéder au système d'exploitation et aux fichiers journaux de l'agent de conteneur Amazon ECS à partir de vos instances de conteneur Amazon ECS. Cette méthode d'accès aux journaux peut être utilisée pour les conteneurs utilisant le type de lancement EC2.
- Amazon CloudWatch Events : associez les événements et acheminez-les vers une ou plusieurs fonctions ou flux cibles afin d'apporter des modifications, de recueillir des informations d'état et de prendre des mesures correctives. Pour plus d'informations, consultez Automatisez les réponses aux erreurs Amazon ECS à l'aide de EventBridge ce guide et découvrez EventBridge l'évolution d'Amazon CloudWatch Events dans le guide de EventBridge l'utilisateur Amazon.
- Container Insights : collectez, agrégez et résumez les métriques et les journaux provenant de vos applications conteneurisées et de vos microservices. Container Insights collecte des données

en tant qu'événements du journal des performances utilisant le format de métrique intégrée. Ces événements du journal des performances sont des entrées qui utilisent un schéma JSON structuré qui permet d'ingérer et de stocker des données à haute cardinalité à grande échelle. À partir de ces données, CloudWatch crée des métriques agrégées au niveau du cluster, de la tâche et du service sous forme de CloudWatch métriques. Les métriques collectées par Container Insights sont disponibles dans des tableaux de bord CloudWatch automatiques et peuvent également être consultées dans la section Metrics de la CloudWatch console.

- AWS CloudTrail surveillance des journaux — Partagez des fichiers journaux entre comptes, surveillez les fichiers CloudTrail journaux en temps réel en les envoyant à CloudWatch Logs, écrivez des applications de traitement des journaux en Java et vérifiez que vos fichiers journaux n'ont pas changé après leur livraison par CloudTrail. Pour plus d'informations, reportez-vous Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail à ce guide et à l'utilisation des fichiers CloudTrail journaux dans le guide de AWS CloudTrail l'utilisateur.
- Surveillance du temps d'exécution : détectez les menaces visant les clusters et les conteneurs au sein de votre AWS environnement. La surveillance du temps d'exécution utilise un agent de GuardDuty sécurité qui augmente la visibilité de l'exécution sur les charges de travail Amazon ECS individuelles, par exemple l'accès aux fichiers, l'exécution des processus et les connexions réseau.

Outils de surveillance manuelle

Un autre aspect important de la surveillance d'Amazon ECS consiste à surveiller manuellement les éléments non couverts par les CloudWatch alarmes. Le CloudWatch tableau de bord de AWS console et les autres tableaux de bord fournissent une at-a-glance vue de l'état de votre AWS environnement. Trusted Advisor Nous vous recommandons de vérifier également les fichiers journaux de vos instances de conteneur et les conteneurs de vos tâches.

- Console Amazon ECS :
  - Métriques du cluster pour le type de EC2 lancement
  - Métriques de service
  - Statut d’intégrité du service
  - Événements de déploiement de services
- CloudWatch page d'accueil :
  - Alarmes et statuts en cours
  - Graphiques des alarmes et des ressources
  - Statut d’intégrité du service

En outre, vous pouvez CloudWatch effectuer les opérations suivantes :

- Créer des tableaux de bord personnalisés pour surveiller les services de votre choix.
- Données de métriques de graphiques pour résoudre les problèmes et découvrir les tendances.
- Recherchez et parcourez tous les indicateurs de vos AWS ressources.
- Créer et Modifier des alarmes pour être informé des problèmes.

- Contrôle de l'état du conteneur : il s'agit de commandes qui s'exécutent localement sur un conteneur et valident l'état et la disponibilité des applications. Vous les configurez par conteneur dans votre définition de tâche.
- AWS Trusted Advisor peut vous aider à surveiller vos AWS ressources afin d'améliorer les performances, la fiabilité, la sécurité et la rentabilité. Quatre Trusted Advisor chèques sont disponibles pour tous les utilisateurs ; plus de 50 chèques sont disponibles pour les utilisateurs disposant d'un plan de support Business ou Enterprise. Pour de plus amples informations, veuillez consulter AWS Trusted Advisor.

  Trusted Advisor possède les vérifications suivantes relatives à Amazon ECS :

  - Tolérance aux pannes qui indique qu'un service fonctionne dans une seule zone de disponibilité.
  - Une tolérance aux pannes qui indique que vous n'avez pas utilisé la stratégie de placement des spreads pour plusieurs zones de disponibilité.
- AWS Compute Optimizer est un service qui analyse les paramètres de configuration et d'utilisation de vos AWS ressources. Il indique si vos ressources sont optimales et génère des recommandations d'optimisation afin de réduire les coûts et d'améliorer les performances de vos charges de travail.

  Pour de plus amples informations, veuillez consulter AWS Compute Optimizer recommandations pour Amazon ECS.

Surveillez Amazon ECS à l'aide de CloudWatch

Vous pouvez surveiller vos ressources Amazon ECS à l'aide d'Amazon CloudWatch, qui collecte et traite les données brutes d'Amazon ECS pour en faire des métriques lisibles en temps quasi réel. Ces statistiques sont enregistrées pour une durée de deux semaines. Vous pouvez, par conséquent, accéder aux informations historiques et mieux comprendre la façon dont vos clusters ou vos services fonctionnent. Les données métriques Amazon ECS sont automatiquement envoyées par CloudWatch

intervalles d'une minute. Pour plus d'informations CloudWatch, consultez le guide de CloudWatch l'utilisateur Amazon.

Amazon ECS fournit des métriques gratuites pour les clusters et les services. Moyennant un coût supplémentaire, vous pouvez activer Amazon ECS CloudWatch Container Insights pour votre cluster pour obtenir des mesures par tâche, notamment l'utilisation du processeur, de la mémoire et du système de fichiers EBS. Pour plus d'informations sur Container Insights, consultez Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée.

Considérations

Les points suivants doivent être pris en compte lors de l'utilisation des CloudWatch métriques Amazon ECS.

- Tous les services Amazon ECS hébergés sur Fargate CloudWatch disposent automatiquement de mesures d'utilisation du processeur et de la mémoire. Vous n'avez donc pas besoin de prendre de mesures manuelles.

- Pour toute tâche ou service Amazon ECS hébergé sur des EC2 instances Amazon, l' EC2 instance Amazon nécessite la version `1.4.0` ou ultérieure (Linux) `1.0.0` ou ultérieure (Windows) de l'agent de conteneur pour que CloudWatch les métriques soient générées. Cependant, nous vous recommandons d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.
- La version minimale de Docker pour des CloudWatch métriques fiables est la version Docker `20.10.13` et les versions plus récentes.
- Vos EC2 instances Amazon nécessitent également l'`ecs:StartTelemetrySession`autorisation du rôle IAM avec lequel vous lancez vos EC2 instances Amazon. Si vous avez créé le rôle IAM de votre instance de conteneur Amazon ECS avant que CloudWatch les métriques ne soient disponibles pour Amazon ECS, vous devrez peut-être ajouter cette autorisation. Pour plus d'informations sur le rôle IAM des instances de conteneur et sur l'attachement de la politique IAM gérée aux instances de conteneur, consultez. Rôle IAM d'instance de conteneur Amazon ECS
- Vous pouvez désactiver la collecte de CloudWatch métriques sur vos EC2 instances Amazon `ECS_DISABLE_METRICS=true` en définissant la configuration de votre agent de conteneur Amazon ECS. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

Métriques recommandées

Amazon ECS fournit des CloudWatch métriques gratuites que vous pouvez utiliser pour surveiller vos ressources. La réservation du processeur et de la mémoire et l'utilisation du processeur, de la mémoire et du système de fichiers EBS dans l'ensemble de votre cluster, ainsi que l'utilisation du processeur, de la mémoire et du système de fichiers EBS sur les services de vos clusters peuvent être mesurées à l'aide de ces métriques. Pour vos charges de travail GPU, vous pouvez mesurer votre réservation GPU sur votre cluster.

L'infrastructure sur laquelle vos tâches Amazon ECS sont hébergées dans vos clusters détermine les métriques disponibles. Pour les tâches hébergées sur l'infrastructure Fargate, Amazon ECS fournit des mesures d'utilisation du processeur, de la mémoire et du système de fichiers EBS afin de faciliter la surveillance de vos services. Pour les tâches hébergées sur EC2 des instances, Amazon ECS fournit des mesures de réservation du processeur, de la mémoire et du processeur graphique, ainsi que des mesures d'utilisation du processeur et de la mémoire au niveau du cluster et du service. Vous devez surveiller séparément les EC2 instances Amazon qui constituent votre infrastructure sous-jacente. Pour plus d'informations sur la surveillance de vos EC2 instances Amazon, consultez la section Monitoring Amazon EC2 dans le guide de EC2 l'utilisateur Amazon.

Pour plus d'informations sur les alarmes recommandées à utiliser avec Amazon ECS, consultez l'une des rubriques suivantes dans le guide de l'utilisateur d'Amazon CloudWatch Logs :

- Amazon ECS
- Amazon ECS avec Container Insights

Affichage des métriques Amazon ECS

Une fois que les ressources sont en cours d'exécution dans votre cluster, vous pouvez consulter les métriques sur Amazon ECS et CloudWatch les consoles. La console Amazon ECS fournit une vue maximum, minimum et moyenne de 24 heures des métriques de votre cluster et du service. La console CloudWatch fournit un affichage personnalisable et précis de vos ressources, ainsi que le nombre de tâches en cours d'exécution dans un service.

Console Amazon ECS

Les métriques d'utilisation des unités UC et de la mémoire Amazon ECS service sont disponibles sur la console Amazon ECS. La vue des métriques de service présente les valeurs moyennes, minimales et maximales pour la période de 24 heures qui précède, avec des points de données disponibles à

intervalle de 5 minutes. Pour de plus amples informations, veuillez consulter Mesures d'utilisation des services Amazon ECS.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sélectionnez le cluster dans lequel vous voulez afficher les métriques.
3. Déterminez les indicateurs à consulter.

Pour consulter les statistiques de	Étapes
Clusters	Sur la page des détails du cluster, choisissez l'onglet Metrics. Un lien vers la CloudWatch console est également fourni pour consulter vos statistiques CloudWatch Container Insights si celles-ci sont activées.
Services	Sur la page des détails du cluster, sous l'onglet Services, sélectionnez le service. Les métriques sont ensuite disponibles dans l'onglet Health and metrics.

CloudWatch console

Pour le type de lancement Fargate, les métriques du service Amazon ECS peuvent également être consultées sur la console. CloudWatch La console fournit la vue la plus détaillée des métriques Amazon ECS et vous pouvez personnaliser les vues en fonction de vos besoins. Vous pouvez consulter l'utilisation du service et le nombre de tâches en cours d'exécution du service.

Pour le type de EC2 lancement, les métriques du cluster et du service Amazon ECS peuvent également être consultées sur la CloudWatch console. La console fournit la vue la plus détaillée des métriques Amazon ECS et vous pouvez personnaliser les vues en fonction de vos besoins.

Pour plus d'informations sur la façon de consulter les statistiques, consultez Afficher les statistiques disponibles dans le guide de CloudWatch l'utilisateur Amazon.

CloudWatch Métriques Amazon ECS

Vous pouvez utiliser les statistiques CloudWatch d'utilisation pour obtenir une visibilité sur l'utilisation des ressources de votre compte. Utilisez ces indicateurs pour visualiser l'utilisation actuelle de vos services sur CloudWatch des graphiques et des tableaux de bord.

Amazon ECS envoie des métriques à des CloudWatch intervalles d'une minute. Ces mesures sont collectées pour les ressources dont les tâches sont définies dans l'`RUNNING`état. Si aucune tâche n'est en cours d'exécution dans un cluster, un service ou une autre ressource, aucune métrique ne sera signalée pour cette ressource pendant cette période. Par exemple, si vous avez un cluster avec un service mais que ce service ne comporte aucune tâche dans un `RUNNING` état, aucune métrique ne sera envoyée à CloudWatch. De même, si vous avez deux services et que l'un d'eux comporte des tâches en cours d'exécution alors que l'autre n'en a pas, seules les mesures relatives au service comportant des tâches en cours d'exécution seront envoyées.

Métrique	Description	Dimension valide	Statistiques utiles	Unit
CPUReservation	Pourcentage d'unités de processeur réservées dans le cluster ou le service. La réservation de processeur (filtrée par`ClusterName`) est mesurée comme le nombre total d'unités de processeur réservées par les	`ClusterName` .	Moyenne, minimale, maximale	Pourcentage

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	tâches Amazon ECS sur le cluster, divisé par le nombre total d'unités de processeur pour toutes les EC2 instances Amazon enregistrées dans le cluster. Seules EC2 les instances Amazon `ACTIVE` ou `DRAINING` leur statut affecteront les métriques de réservation du processeur. La métrique n'est prise en charge que pour les tâches hébergées sur une EC2 instance Amazon.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`CPUUtilization`	Pourcentage d'unités de processeur utilisées par le cluster ou le service. L'utilisation du processeur au niveau du cluster (filtrée par`ClusterName`) est mesurée comme le nombre total d'unités de processeur utilisées par les tâches Amazon ECS sur le cluster, divisé par le nombre total d'unités de processeur pour toutes les EC2 instances Amazon enregistrées dans le cluster. Seules EC2 les instances Amazon `ACTIVE` ou `DRAINING` leur statut	`ClusterName` , `ServiceName`	Moyenne, minimale, maximale	Pourcentage

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	affecteront les métriques de réservation du processeur. La métrique au niveau du cluster n'est prise en charge que pour les tâches hébergées sur une instance Amazon. EC2 L'utilisation du processeur au niveau du service (filtrée par`ClusterName`,`ServiceName`) est mesurée comme le nombre total d'unités de processeur utilisées par les tâches appartenant au service, divisé par le nombre total d'unités de processeur réservées aux tâches appartenant			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	au service. La métrique de niveau de service est prise en charge pour les tâches hébergées sur les EC2 instances Amazon et Fargate.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`MemoryReservation`	Pourcentage de mémoire qui est réservé en exécutant des tâches dans le cluster. La réserve de mémoire du cluster est mesurée comme la mémoire totale réservée par les tâches Amazon ECS sur le cluster, divisée par la quantité totale de mémoire pour toutes les EC2 instances Amazon enregistrées dans le cluster. Cette métrique ne peut être filtrée que par`ClusterName` . Seules EC2 les instances `ACTIVE` ou le `DRAINING` statut d'Amazon affecteront les	`ClusterName` .	Moyenne, minimale, maximale	Pourcentage

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	métriques de réservation de mémoire. La métrique de réservation de mémoire au niveau du cluster n'est prise en charge que pour les tâches hébergées sur une EC2 instance Amazon. Note Lors du calcul de l'utilisation de la mémoire, si elle `MemoryReservation` est spécifiée, elle est utilisée dans le calcul au lieu de la			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	mémoire totale.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`MemoryUtilization`	Pourcentage de mémoire utilisé par le cluster ou le service. L'utilisation de la mémoire au niveau du cluster (filtrée par`ClusterName`) est mesurée comme la mémoire totale utilisée par les tâches Amazon ECS sur le cluster, divisée par la mémoire totale de toutes les EC2 instances Amazon enregistrées dans le cluster. Seules EC2 les instances Amazon dans `ACTIVE` ou `DRAINING` leur statut auront une incidence sur les mesures d'utilisation de la mémoire. La	`ClusterName` , `ServiceName`	Moyenne, minimale, maximale	Pourcentage

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	métrique au niveau du cluster n'est prise en charge que pour les tâches hébergées sur une instance Amazon. EC2 L'utilisation de la mémoire au niveau du service (filtrée par`ClusterName`,`ServiceName`) est mesurée comme la mémoire totale utilisée par les tâches appartenant au service, divisée par la mémoire totale réservée aux tâches appartenant au service. La métrique de niveau de service est prise en charge pour les tâches hébergées sur les EC2			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	instances Amazon et Fargate.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`EBSFilesystemUtilization`	Pourcentage du système de fichiers Amazon EBS utilisé par les tâches d'un service. La métrique d'utilisation du système de fichiers EBS au niveau du service (filtrée par`ClusterName` ,`ServiceName`) est mesurée comme la quantité totale du système de fichiers EBS utilisée par les tâches appartenant au service, divisée par la quantité totale de stockage du système de fichiers EBS allouée à toutes les tâches appartenant au service. La métrique d'utilisa	`ClusterName` , `ServiceName`	Moyenne, minimale, maximale	Pourcentage

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	tion du système de fichiers EBS au niveau de service n'est disponible que pour les tâches hébergées sur des EC2 instances Amazon (à l'aide de la version de l'agent de conteneur `1.79.0`) et sur Fargate (à l'aide de la version de plate-forme`1.4.0`) auxquelles un volume EBS est attaché. **Note** Pour les tâches hébergées sur Fargate, il y a de l'espace sur le disque uniquemen			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	t utilisé par Fargate. L'espace utilisé par Fargate est gratuit, mais vous pourrez voir ce stockage supplémentaire à l'aide d'outils tels que. `df`			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`GPUReservation`	Pourcentage du total disponible GPUs réservé en exécutant des tâches dans le cluster. La métrique de réservation du GPU au niveau du cluster est mesurée comme le nombre de tâches GPUs réservées par Amazon ECS sur le cluster, divisé par le nombre total de tâches disponibles sur toutes les EC2 instances Amazon GPUs enregistrées dans le cluster. GPUs Seules EC2 les instances Amazon présentes `ACTIVE` ou `DRAINING` leur statut affecteront les métriques de	`ClusterName`	Moyenne, minimale, maximale	Pourcentage

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	réservation du GPU.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`ActiveConnectionCount`	Nombre total de connexions simultanées actives entre les clients et les proxys Amazon ECS Service Connect qui s'exécutent dans le cadre de tâches qui partagent le `DiscoveryName` sélectionné. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect. Dimensions valides : `DiscoveryName` et `DiscoveryName, ServiceName, ClusterName`.	`DiscoveryName` et `DiscoveryName ServiceName`, `ClusterName`	Moyenne, minimum, maximum, somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`NewConnectionCount`	Nombre total de nouvelles connexions établies entre les clients et les proxys Service Connect Amazon ECS qui s'exécutent dans le cadre de tâches qui partagent le `Discovery Name` sélectionné. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`Discovery Name` et `Discovery Name, ServiceName, ClusterName`	Moyenne, minimum, maximum, somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`ProcessedBytes`	Nombre total d'octets de trafic entrant traités par les proxys Service Connect. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`DiscoveryName` et `DiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, minimum, maximum, somme	Octets

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`RequestCount`	Nombre de demandes de trafic entrant traitées par les proxys Service Connect. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect. Vous devez également configurer `appProtocol` le mappage des ports dans votre définition de tâche.	`DiscoveryName` et `DiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, minimum, maximum, somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`GrpcRequestCount`	Nombre de demandes de trafic entrant gRPC traitées par les proxys Service Connect. Cette métrique n'est disponible que si vous avez configuré Amazon ECS Service Connect et que le `appProtocol` est `GRPC` dans le mappage des ports de la définition de la tâche.	`DiscoveryName` et `DiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, minimum, maximum, somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`HTTPCode_Target_2XX_Count`	Nombre de codes de réponse HTTP numérotés de 200 à 299 générés par les applications dans le cadre de ces tâches. Ces tâches sont les cibles. Cette métrique ne compte que les réponses envoyées aux proxys Service Connect par les applications dans le cadre de ces tâches, et non les réponses envoyées directement. Cette métrique n'est disponible que si vous avez configuré Amazon ECS Service Connect et que le `appProtocol` est `HTTP` ou `HTTP2` dans	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, minimum, maximum, somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	le mappage des ports de la définition de la tâche. Dimensions valides :.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`HTTPCode_Target_3XX_Count`	Nombre de codes de réponse HTTP numérotés de 300 à 399 générés par les applications dans le cadre de ces tâches. Ces tâches sont les cibles. Cette métrique ne compte que les réponses envoyées aux proxys Service Connect par les applications dans le cadre de ces tâches, et non les réponses envoyées directement. Cette métrique n'est disponible que si vous avez configuré Amazon ECS Service Connect et que le `appProtocol` est `HTTP` ou `HTTP2` dans	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, minimum, maximum, somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	le mappage des ports de la définition de la tâche.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`HTTPCode_Target_4XX_Count`	Nombre de codes de réponse HTTP numérotés de 400 à 499 générés par les applications dans le cadre de ces tâches. Ces tâches sont les cibles. Cette métrique ne compte que les réponses envoyées aux proxys Service Connect par les applications dans le cadre de ces tâches, et non les réponses envoyées directement. Cette métrique n'est disponible que si vous avez configuré Amazon ECS Service Connect et que le `appProtocol` est `HTTP` ou `HTTP2` dans	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, minimum, maximum, somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	le mappage des ports de la définition de la tâche.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`HTTPCode_Target_5XX_Count`	Nombre de codes de réponse HTTP numérotés de 500 à 599 générés par les applications dans le cadre de ces tâches. Ces tâches sont les cibles. Cette métrique ne compte que les réponses envoyées aux proxys Service Connect par les applications dans le cadre de ces tâches, et non les réponses envoyées directement. Cette métrique n'est disponible que si vous avez configuré Amazon ECS Service Connect et que le `appProtocol` est `HTTP` ou `HTTP2` dans	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, minimum, maximum, somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	le mappage des ports de la définition de la tâche.			
`RequestCountPerTarget`	Nombre moyen de demandes reçues par chaque cible qui partage le `Discovery Name` sélectionné. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne	Nombre
`TargetProcessedBytes`	Nombre total d'octets traités par les proxys Service Connect. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, minimum, maximum, somme	Octets

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`TargetResponseTime`	Latence du traitement des demandes d'application. Temps écoulé, en millisecondes, entre le moment où la demande arrive au proxy Service Connect dans la tâche cible et le moment où une réponse de l'application cible arrive au proxy. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, minimale, maximale	Millisecondes

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`ClientTLSNegotiationErrorCount`	Nombre total de fois où la connexion TLS a échoué. Cette métrique n'est utilisée que lorsque le protocole TLS est activé. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`DiscoveryName` et `DiscoveryName ServiceName` , `ClusterName`	Moyenne, minimum, maximum, somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`TargetTLSNegotiationErrorCount`	Nombre total de fois où la connexion TLS a échoué en raison de certificats clients manquants, d'échecs de AWS Private CA vérifications ou de vérifications SAN infructueuses. Cette métrique n'est utilisée que lorsque le protocole TLS est activé. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`ServiceName` ,`ClusterName` , `TargetDiscoveryName` et `TargetDiscoveryName`	Moyenne, minimum, maximum, somme	Nombre

Dimensions pour les métriques Amazon ECS

Les métriques Amazon ECS utilisent l'espace de noms `AWS/ECS` et fournissent des métriques pour les dimensions suivantes. Amazon ECS envoie uniquement des métriques pour les ressources dont les tâches sont à l'état `RUNNING`. Par exemple, si vous avez un cluster avec un service mais que ce service n'a aucune tâche avec l'état `RUNNING`, aucune métrique n'est envoyée à CloudWatch.

Si vous disposez de deux services et que l'un d'eux seulement a des tâches en cours d'exécution, seules les métriques du service avec des tâches en cours d'exécution seront envoyées.

Dimension	Définition
`ClusterName`	Cette dimension filtre les données que vous demandez pour toutes les ressources dans un cluster donné. Toutes les métriques Amazon ECS sont filtrées par `ClusterName` .
`ServiceName`	Cette dimension filtre les données que vous demandez pour toutes les ressources dans un service spécifié au sein d'un cluster spécifié.
`DiscoveryName`	Cette dimension filtre les données que vous demandez pour les métriques de trafic vers un nom de découverte Service Connect spécifié dans tous les clusters Amazon ECS. Notez qu'un port spécifique d'un conteneur en cours d'exécution peut avoir plusieurs noms de découverte.
`DiscoveryName, ServiceName, ClusterName`	Cette dimension filtre les données que vous demandez pour les métriques de trafic vers un nom de découverte Service Connect spécifié pour les tâches portant ce nom de découverte et créées par ce service dans ce cluster. Utilisez cette dimension pour voir les métriques du trafic entrant d'un service spécifique, si vous avez réutilisé le même nom de découverte dans plusieurs services dans des espaces de noms différents. Notez qu'un port spécifique d'un conteneur en cours d'exécution peut avoir plusieurs noms de découverte.

Dimension	Définition
`TargetDiscoveryName`	Cette dimension filtre les données que vous demandez pour les métriques de trafic vers un nom de découverte Service Connect spécifié dans tous les clusters Amazon ECS. À la différence de `DiscoveryName` , ces métriques de trafic mesurent uniquement le trafic entrant vers ce `DiscoveryName` issu d'autres tâches Amazon ECS ayant une configuration Service Connect dans cet espace de noms. Cela inclut les tâches effectuées par les services avec une configuration Service Connect client uniquement ou client-serveur. Notez qu'un port spécifique d'un conteneur en cours d'exécution peut avoir plusieurs noms de découverte.

Dimension	Définition
`TargetDiscoveryName, ServiceName, ClusterName`	Cette dimension filtre les données que vous demandez pour les métriques de trafic vers un nom de découverte Service Connect spécifié, mais ne compte que le trafic des tâches créées par ce service dans ce cluster. Utilisez cette dimension pour voir les métriques de trafic entrant provenant d'un client spécifique dans un autre service. À la différence de `DiscoveryName, ServiceName, ClusterName` , ces métriques de trafic mesurent uniquement le trafic entrant vers ce `DiscoveryName` issu d'autres tâches Amazon ECS ayant une configuration Service Connect dans cet espace de noms. Cela inclut les tâches effectuées par les services avec une configuration Service Connect client uniquement ou client-serveur. Notez qu'un port spécifique d'un conteneur en cours d'exécution peut avoir plusieurs noms de découverte.

AWS Fargate métriques d'utilisation

Vous pouvez utiliser les statistiques CloudWatch d'utilisation pour obtenir une visibilité sur l'utilisation des ressources de votre compte. Utilisez ces indicateurs pour visualiser l'utilisation actuelle de vos services sur CloudWatch des graphiques et des tableaux de bord.

AWS Fargate les métriques d'utilisation correspondent aux quotas AWS de service. Vous pouvez configurer des alarmes qui vous alertent lorsque votre utilisation approche d’un quota de service. Pour plus d'informations sur les quotas de service Fargate, les points de terminaison Amazon ECS et les quotas dans le... Référence générale d'Amazon Web Services

AWS Fargate publie les métriques suivantes dans l'espace de `AWS/Usage` noms.

Métrique	Description
`ResourceCount`	Nombre total des ressources spécifiées exécutées sur votre compte. La ressource est définie par les dimensions associées à la métrique.

Les dimensions suivantes permettent d'affiner les métriques d'utilisation publiées par AWS Fargate.

Dimension	Description
`Service`	Nom du AWS service contenant la ressource. Pour les métriques d'utilisation d' AWS Fargate , la valeur de cette dimension est `Fargate`.
`Type`	Type d'entité faisant l'objet d'un rapport. Actuellement, la seule valeur valide pour les métriques AWS Fargate d'utilisation est`Resource`.
`Resource`	Type de ressource en cours d'exécution. Type de ressource en cours d'exécution. Actuellement, la seule valeur valide pour les métriques AWS Fargate d'utilisation est celle `vCPU` qui renvoie des informations sur les instances en cours d'exécution.
`Class`	Classe de ressource suivie. Classe de ressource suivie. Pour les métriques AWS Fargate d'utilisation avec vCPU comme valeur de la dimension Resource, les valeurs valides sont `Standard/OnDemand` et. `Standard/Spot`

Vous pouvez utiliser la console Service Quotas pour visualiser votre utilisation sur un graphique et configurer des alarmes qui vous alertent lorsque votre AWS Fargate utilisation approche d'un quota de service. Pour plus d'informations sur la façon de créer une CloudWatch alarme pour vous avertir lorsque vous approchez d'un seuil de quota, consultez les sections Service Quotas et Amazon CloudWatch alarmes dans le guide de Quotas de service l'utilisateur

.

Métriques de réservation du cluster Amazon ECS

Les métriques de réservation de cluster sont mesurées en pourcentage du processeur, de la mémoire et de GPUs ce qui est réservé par toutes les tâches Amazon ECS sur un cluster par rapport au processeur et GPUs à la mémoire agrégés enregistrés pour chaque instance de conteneur active du cluster. Seules les instances de conteneur ayant le statut `ACTIVE` ou `DRAINING` affectent les métriques de réservation de cluster. Cette métrique est utilisée uniquement sur les clusters dont les tâches ou les services sont hébergés sur EC2 des instances. Il n'est pas pris en charge sur les clusters sur lesquels les tâches sont hébergées AWS Fargate.

```
                                (Total CPU units reserved by tasks in cluster) x 100
Cluster CPU reservation =
 ---------------------------------------------------------------
                           (Total CPU units registered by container instances in
 cluster)
```

```
                                  (Total MiB of memory reserved by tasks in cluster x
 100)
Cluster memory reservation =
 ------------------------------------------------------------------
                              (Total MiB of memory registered by container instances in
 cluster)
```

```
                                 (Total GPUs reserved by tasks in cluster x 100)
Cluster GPU reservation =
 -------------------------------------------------------------------
                              (Total GPUs registered by container instances in cluster)
```

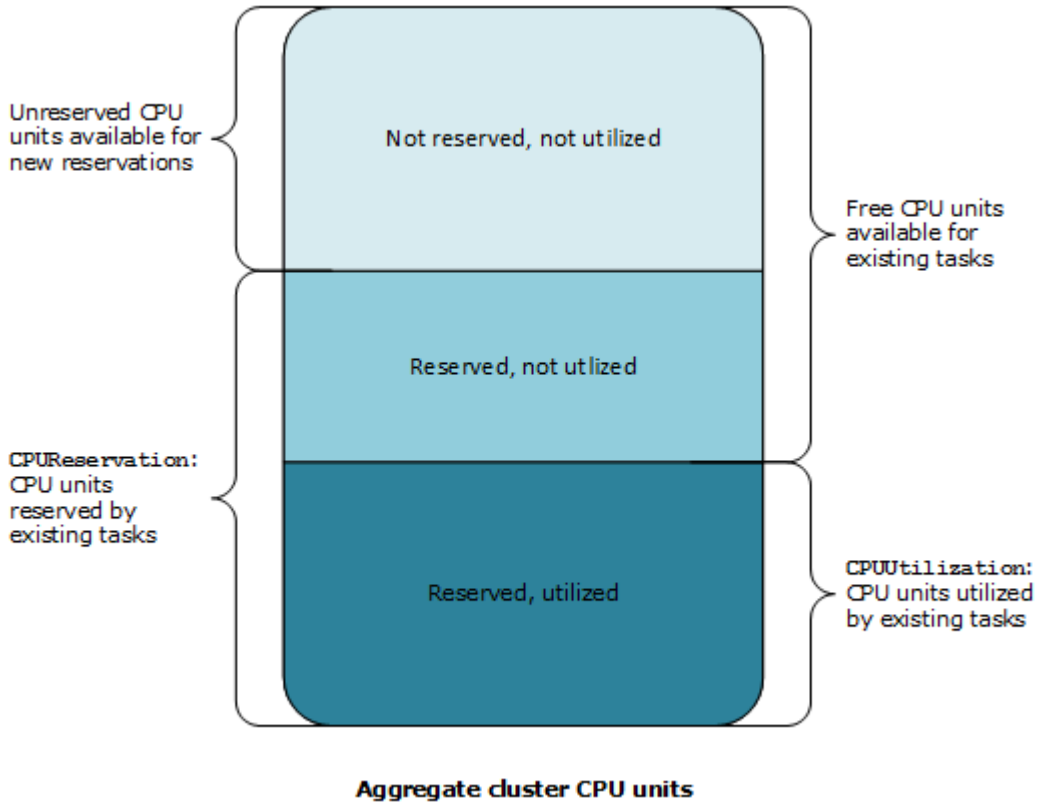
Lorsque vous exécutez une tâche dans un cluster, Amazon ECS analyse sa définition de tâche et réserve les unités de processeur agrégées, les MiB de mémoire GPUs et celles spécifiées dans ses définitions de conteneur. Chaque minute, Amazon ECS calcule le nombre d'unités de processeur, de MiB de mémoire, GPUs actuellement réservées à chaque tâche exécutée dans le cluster. La quantité totale de CPU, de mémoire et de mémoire GPUs réservée à toutes les tâches exécutées sur le cluster est calculée, et ces chiffres sont présentés CloudWatch sous forme de pourcentage du total des ressources enregistrées pour le cluster. Si vous spécifiez une limite flexible (`memoryReservation`) dans la définition de tâche, elle est utilisée pour calculer la quantité de mémoire réservée. Sinon, la limite stricte (`memory`) est utilisée. Le total des Mio de mémoire réservés par les tâches d'un cluster inclut également la taille du volume du système de fichiers temporaire

(`tmpfs`), et `sharedMemorySize`, si elle est définie dans la définition de la tâche. Pour en savoir plus sur les limites strictes et flexibles, la taille de mémoire partagée et la taille de volume tmpfs, veuillez consulter Paramètres de définition de tâche (langue française non garantie).

Prenons l'exemple d'un cluster qui possède deux instances de conteneur actives enregistrées : une instance `c4.4xlarge` et une instance `c4.large`. L'instance `c4.4xlarge` s'enregistre dans le cluster avec 16 384 unités d'UC et 30 158 Mio de mémoire. L'instance `c4.large` s'enregistre avec 2 048 unités d'UC et 3 768 Mio de mémoire. Les ressources globales de ce cluster sont de 18 432 unités d'UC et de 33 926 Mio de mémoire.

Si une définition de tâche se réserve 1 024 unités d'UC et 2 048 Mio de mémoire, et que dix tâches sont lancées avec cette définition de tâche sur ce cluster (et qu'aucune autre tâche n'est en cours d'exécution), un total de 10 240 unités d'UC et 20 480 Mio de mémoire est réservé. Cela correspond à 55 % CloudWatch de réservation du processeur et à 60 % de réservation de mémoire pour le cluster.

L'illustration suivante présente le nombre total d'unités d'UC enregistrées dans un cluster et la signification de leur réservation et de leur utilisation pour les tâches existantes et le placement des nouvelles tâches. Les blocs inférieur (réservé, utilisé) et central (réservé, non utilisé) représentent le nombre total d'unités de processeur réservées aux tâches existantes exécutées sur le cluster, ou la `CPUReservation` CloudWatch métrique. Le bloc inférieur représente les unités de processeur réservées que les tâches en cours d'exécution utilisent réellement sur le cluster, ou la `CPUUtilization` CloudWatch métrique. Le bloc supérieur représente les unités d'UC qui ne sont pas réservées par des tâches existantes. Ces unités d'UC sont disponibles pour le placement de nouvelles tâches. Les tâches existantes peuvent également utiliser ces unités d'UC non réservées, si leur besoin en ressources processeur augmente. Pour plus d'informations, consultez la documentation relative aux paramètres de définition de tâche `cpu`.



Mesures d'utilisation du cluster Amazon ECS

Les mesures d'utilisation du cluster sont disponibles pour le processeur, la mémoire et, lorsqu'un volume EBS est associé à vos tâches, l'utilisation du système de fichiers EBS. Ces métriques ne sont disponibles que pour les clusters dont les tâches ou les services sont hébergés sur EC2 des instances Amazon. Ils ne sont pas pris en charge sur les clusters sur lesquels les tâches sont hébergées AWS Fargate.

Mesures d'utilisation du processeur et de la mémoire au niveau du cluster Amazon ECS

L'utilisation du processeur et de la mémoire est mesurée comme le pourcentage du processeur et de la mémoire utilisés par toutes les tâches d'un cluster par rapport au processeur et à la mémoire agrégés enregistrés pour chaque EC2 instance Amazon active enregistrée dans le cluster. Seules EC2 les instances Amazon en `DRAINING` statut `ACTIVE` ou en statut affecteront les métriques d'utilisation du cluster.

```
                                  (Total CPU units used by tasks in cluster) x 100
Cluster CPU utilization =
 ---------------------------------------------------------------
```

```
                            (Total CPU units registered by container instances in
 cluster)
```

```
                                      (Total MiB of memory used by tasks in cluster x
 100)
Cluster memory utilization =
 ------------------------------------------------------------------
                                 (Total MiB of memory registered by container instances in
 cluster)
```

Chaque minute, l'agent de conteneur Amazon ECS de chaque EC2 instance Amazon calcule le nombre d'unités CPU et de MiB de mémoire actuellement utilisés pour chaque tâche exécutée sur cette instance EC2 Amazon, et ces informations sont transmises à Amazon ECS. La quantité totale de CPU et de mémoire utilisée pour toutes les tâches exécutées sur le cluster est calculée, et ces chiffres sont présentés CloudWatch sous forme de pourcentage du total des ressources enregistrées pour le cluster.

Par exemple, un cluster possède deux EC2 instances Amazon actives enregistrées, une `c4.4xlarge` instance et une `c4.large` instance. L'`c4.4xlarge`instance s'enregistre dans le cluster avec des unités `16,384` CPU et des `30,158` MiB de mémoire. L'`c4.large`instance s'enregistre avec les unités de `2,048` processeur et les `3,768` Mio de mémoire. Les ressources agrégées de ce cluster sont les unités `18,432` CPU et les `33,926` MiB de mémoire.

Si dix tâches sont exécutées sur ce cluster et que chaque tâche consomme des unités de `1,024` processeur et des `2,048` Mo de mémoire, un total d'unités de `10,240` processeur et de `20,480` Mo de mémoire sont utilisés sur le cluster. Cela correspond à 55 % CloudWatch d'utilisation du processeur et à 60 % d'utilisation de la mémoire pour le cluster.

Utilisation du système de fichiers Amazon EBS au niveau du cluster Amazon ECS

La métrique d'utilisation du système de fichiers EBS au niveau du cluster est mesurée comme la quantité totale du système de fichiers EBS utilisée par les tâches exécutées sur le cluster, divisée par la quantité totale de stockage du système de fichiers EBS allouée à toutes les tâches du cluster.

```
                                        (Total GB of EBS filesystem used by tasks in
 cluster x 100)
Cluster EBS filesystem utilization =
 ----------------------------------------------------------------
                                        (Total GB of EBS filesystem allocated to tasks
 in cluster)
```

Mesures d'utilisation des services Amazon ECS

Les mesures d'utilisation du service sont disponibles pour le processeur, la mémoire et, lorsqu'un volume EBS est associé à vos tâches, l'utilisation du système de fichiers EBS. Les métriques de niveau de service sont prises en charge pour les services dont les tâches sont hébergées à la fois sur les EC2 instances Amazon et sur Fargate.

Niveau de service : utilisation du processeur et de la mémoire

L'utilisation du processeur et de la mémoire est mesurée comme le pourcentage du processeur et de la mémoire utilisés par les tâches Amazon ECS appartenant à un service sur un cluster par rapport au processeur et à la mémoire spécifiés dans la définition des tâches du service.

Lorsque vous consultez ces statistiques dans CloudWatch, vous pouvez choisir différentes statistiques :

- Moyenne : taux d'utilisation moyen pour toutes les tâches du service. Ce montant est calculé à l'aide de la formule ci-dessous.
- Minimum : utilisation de la tâche dont l'utilisation des ressources est la plus faible dans le service. Cela représente le pourcentage de processeur ou de mémoire utilisé par la tâche la moins gourmande en ressources par rapport à ce qui a été spécifié dans la définition de la tâche.
- Maximum : utilisation de la tâche utilisant le plus de ressources dans le service. Cela représente le pourcentage de processeur ou de mémoire utilisé par la tâche la plus gourmande en ressources par rapport à ce qui a été spécifié dans la définition de la tâche.

Les formules suivantes montrent comment la statistique moyenne est calculée :

```
                                  (Total CPU units used by tasks in service) x 100
Service CPU utilization =
 ----------------------------------------------------------------
                           (Total CPU units specified in the task definition) x (Number
 of tasks in the service)
```

```
                                     (Total MiB of memory used by tasks in service x
 100)
Service memory utilization =
 -------------------------------------------------------------------
                              (Total MiB of memory specified in the task definition)  x
 (Number of tasks in the service)
```

> **Note**
>
> Les formules ci-dessus s'appliquent uniquement à la statistique moyenne. Les statistiques minimale et maximale représentent la tâche individuelle dont l'utilisation des ressources est la plus faible et la plus élevée, respectivement, plutôt qu'un calcul agrégé pour toutes les tâches.

Amazon ECS collecte des métriques toutes les 20 secondes. Chaque minute, l'agent de conteneur Amazon ECS calcule le nombre d'unités CPU et de MiB de mémoire actuellement utilisés pour chaque tâche en cours d'exécution appartenant au service. Ces informations sont transmises à Amazon ECS. La quantité totale de ressources processeur et mémoire utilisées par toutes les tâches du service en cours d'exécution sur le cluster est calculée, et ces chiffres sont transmis à CloudWatch sous la forme d'un pourcentage du total des ressources spécifiées pour le service dans la définition de tâche du service. Les valeurs minimale et maximale sont les plus petites et les plus grandes des métriques de 20 secondes. La moyenne est l'agrégat des 3 valeurs.

Si vous spécifiez une limite flexible (`memoryReservation`), elle est utilisée pour calculer la quantité de mémoire réservée. Sinon, la limite stricte (`memory`) est utilisée. Pour plus d'informations sur les limites strictes et souples, consultezTaille de la tâche.

Prenons l'exemple d'un service pour lequel la définition de tâche spécifie un total de 512 unités d'UC et 1 024 Mio de mémoire (avec le paramètre de limite stricte `memory`) pour l'ensemble de ses conteneurs. Le service a un nombre souhaité de 1 tâche en cours d'exécution, il s'exécute sur un cluster avec 1 instance de conteneur `c4.large` (avec 2 048 unités d'UC et 3 768 Mio de mémoire totale) et il n'y a pas d'autre tâche en cours d'exécution sur le cluster. Bien que la tâche spécifie 512 unités d'UC, comme il s'agit de la seule tâche en cours d'exécution sur une instance de conteneur comportant 2 048 unités d'UC, elle peut utiliser jusqu'à quatre fois la quantité spécifiée (2 048/512). Toutefois, la mémoire spécifiée de 1 024 Mio est une limite stricte et elle ne peut pas être dépassée. Ainsi, dans ce cas, l'utilisation de la mémoire par le service ne peut pas dépasser 100 %.

Si l'exemple précédent utilise la limite flexible `memoryReservation` au lieu du paramètre de limite stricte `memory`, les tâches du service peuvent, au besoin, utiliser plus de mémoire que les 1 024 Mio spécifiés. Dans ce cas, l'utilisation de la mémoire du service peut dépasser 100 %.

Si votre application connaît un pic soudain d'utilisation de la mémoire pendant une courte période, vous ne verrez pas l'utilisation de la mémoire du service augmenter, car Amazon ECS collecte

plusieurs points de données par minute, puis les agrège en un point de données à CloudWatch qui est envoyé.

Si cette tâche utilise intensivement l'UC à un moment donné et utilise la totalité des 2 048 unités d'UC disponibles et 512 Mio de mémoire, le service signale 400 % d'utilisation de l'UC et 50 % d'utilisation de la mémoire. Si la tâche est inactive et utilise 128 unités d'UC et 128 Mio de mémoire, le service signale 25 % d'utilisation de l'UC et 12,5 % d'utilisation de la mémoire.

> **Note**
>
> Dans cet exemple, l'utilisation de l'UC ne dépasse 100 % que lorsque les unités du processeur sont définies au niveau du conteneur. Si vous définissez des unités d'UC au niveau de la tâche, l'utilisation ne dépasse pas la limite définie au niveau de la tâche.

Utilisation du système de fichiers EBS au niveau de service

L'utilisation du système de fichiers EBS au niveau du service est mesurée comme la quantité totale du système de fichiers EBS utilisée par les tâches appartenant au service, divisée par la quantité totale de stockage du système de fichiers EBS allouée à toutes les tâches appartenant au service.

Nombre de tâches **`RUNNING`** dans le service

Vous pouvez utiliser CloudWatch des métriques pour afficher le nombre de tâches de vos services qui sont dans l'`RUNNING`état. Par exemple, vous pouvez définir une CloudWatch alarme pour cette métrique afin de vous avertir si le nombre de tâches en cours dans votre service tombe en dessous d'une valeur spécifiée.

Nombre de **`RUNNING`** tâches de service dans Amazon ECS CloudWatch Container Insights

Une métrique « Nombre de tâches en cours » (`RunningTaskCount`) est disponible par cluster et par service lorsque vous utilisez Amazon ECS CloudWatch Container Insights. Vous pouvez utiliser Container Insights pour tous les nouveaux clusters créés en activant les paramètres du `containerInsights` compte, sur des clusters individuels en activant les paramètres du cluster lors de la création du cluster, ou sur des clusters existants en utilisant l' UpdateClusterSettings API. Les métriques collectées par CloudWatch Container Insights sont facturées en tant que métriques personnalisées. Pour plus d'informations sur la tarification CloudWatch , consultez Tarification CloudWatch.

Pour consulter cette statistique, consultez les métriques Amazon ECS Container Insights dans le guide de CloudWatch l'utilisateur Amazon.

Cas d'utilisation des métriques d'utilisation des services Amazon ECS

La liste suivante fournit des informations sur les circonstances dans lesquelles utiliser les métriques Amazon ECS.

- Surveillance de l'utilisation des ressources : utilisez des statistiques moyennes pour surveiller les modèles de consommation du processeur et de la mémoire, établir des bases de performance et détecter les tendances de dégradation progressive des performances.
- Optimisation des coûts : utilisez des statistiques moyennes pour surveiller l'utilisation des ressources, dimensionner correctement les conteneurs, ajuster les réservations en fonction des modèles d'utilisation et mettre en œuvre un dimensionnement planifié.
- Analyse comparative des performances : utilisez des statistiques moyennes pour comparer les indicateurs entre les révisions des services, établir les performances KPIs et valider les améliorations d'optimisation.
- Détection du plancher de ressources : utilisez des statistiques moyennes pour identifier les besoins en ressources minimales pendant les périodes d'inactivité, définir les réservations appropriées et détecter les baisses anormales.
- Détection des anomalies : utilisez des statistiques minimales pour détecter les baisses inhabituelles d'utilisation des ressources qui indiquent des problèmes potentiels, tels que des échecs d'initialisation ou des périodes d'inactivité inattendues.
- Affinement de la politique de dimensionnement : utilisez des statistiques minimales pour établir des seuils de mise à l'échelle optimaux sur la base d'une utilisation viable minimale afin d'éviter une mise à l'échelle agressive.
- Planification des capacités : utilisez un maximum de statistiques pour définir des tailles de tâches appropriées et planifier l'infrastructure avec une marge de manœuvre suffisante pour faire face aux pics de trafic.
- Identification des goulots d'étranglement en matière de performances : utilisez un maximum de statistiques pour détecter les points de saturation des ressources, identifier les goulots d'étranglement et déterminer quand augmenter la taille des tâches.
- Configuration de la politique de dimensionnement : utilisez des statistiques maximales pour définir des seuils de scale-out optimaux en fonction des modèles de pic et configurer la capacité de rafale.

- Surveillance de la conformité aux SLA : utilisez un maximum de statistiques pour suivre les temps de réponse de pointe et les taux d'erreur afin de garantir que les performances du service répondent aux exigences définies SLAs.

Informations sur les métriques associées

Pour plus d'informations sur Container Insights, consultez la section Cas d'utilisation de Container Insights dans le guide de CloudWatch l'utilisateur.

Automatisez les réponses aux erreurs Amazon ECS à l'aide de EventBridge

Amazon EventBridge vous permet d'automatiser vos AWS services et de répondre automatiquement aux événements du système tels que les problèmes de disponibilité des applications ou les modifications des ressources. Les événements AWS liés aux services sont diffusés EventBridge en temps quasi réel. Vous pouvez écrire des règles simples pour préciser les événements qui vous intéressent et les actions automatisées à effectuer quand un événement correspond à une règle. Les actions pouvant être configurées automatiquement sont les suivantes :

- Ajouter des événements à des groupes de CloudWatch journaux dans Logs
- Invoquer une fonction AWS Lambda
- Invocation de la commande Amazon EC2 Run
- Relais de l'événement à Amazon Kinesis Data Streams
- Activation d'une machine à AWS Step Functions états
- Notification d'une rubrique Amazon SNS ou d'une file d'attente Amazon Simple Queue Service (Amazon SQS)

Pour plus d'informations, consultez Getting started with Amazon EventBridge dans le guide de EventBridge l'utilisateur Amazon.

Vous pouvez utiliser les événements Amazon ECS pour EventBridge recevoir des notifications en temps quasi réel concernant l'état actuel de vos clusters Amazon ECS. Si vos tâches utilisent le type de EC2 lancement, vous pouvez voir l'état des instances de conteneur et l'état actuel de toutes les tâches exécutées sur ces instances de conteneur. Si vos tâches utilisent le type de lancement Fargate, vous pouvez voir l'état des instances de conteneur.

Vous pouvez ainsi créer des planificateurs personnalisés sur Amazon ECS EventBridge, chargés d'orchestrer les tâches entre les clusters et de surveiller l'état des clusters en temps quasi réel. Vous pouvez éliminer le code de planification et de surveillance qui interroge en permanence le service Amazon ECS pour détecter les changements de statut et gérer les changements d'état Amazon ECS de manière asynchrone en utilisant n'importe quelle EventBridge cible. Les cibles peuvent inclure AWS Lambda Amazon Simple Queue Service, Amazon Simple Notification Service ou Amazon Kinesis Data Streams.

Un flux d'événements Amazon ECS garantit que chaque événement est diffusé au moins une fois. Si des événements dupliqués sont envoyés, l'événement fournit suffisamment d'informations pour identifier les doublons. Pour de plus amples informations, veuillez consulter Gestion des événements Amazon ECS.

Les événements sont relativement classés, afin que vous puissiez facilement savoir lorsqu'un événement s'est produit en rapport avec d'autres événements.

Rubriques

- Événements Amazon ECS
- Gestion des événements Amazon ECS

Événements Amazon ECS

Amazon ECS suit l'état de chacune de vos tâches et de chacun de vos services. Si l'état d'une tâche ou d'un service change, un événement est généré et envoyé à Amazon EventBridge. Ces événements sont considérés comme des événements de changement d'état de tâche et des événements d'action de service. Ces événements et leurs causes possibles sont détaillés dans les sections suivantes.

Amazon ECS génère et envoie les types d'événements suivants à EventBridge :

- Modification de l'état de l'instance de conteneur
- Modification de l'état de la tâche
- Deployment state change (Changement d'état du déploiement)
- Action de service

> **Note**
>
> Amazon ECS pourrait ajouter d'autres types d'événements, sources et informations à l'avenir. Si vous désérialisez des données JSON d'événements dans du code, assurez-vous que votre application est prête à gérer les propriétés inconnues afin d'éviter les problèmes si et quand ces propriétés supplémentaires sont ajoutées.

Dans certains cas, plusieurs événements sont générés pour la même activité. Par exemple, lorsqu'une tâche est démarrée sur une instance de conteneur, un événement de modification de l'état de tâche est généré pour la nouvelle tâche. Un événement de changement d'état d'instance de conteneur est généré pour prendre en compte le changement dans les ressources disponibles, comme le processeur, la mémoire et les ports disponibles, dans l'instance de conteneur. De même, si une instance de conteneur est mise hors service, des événements sont générés pour l'instance de conteneur, l'état de connexion de l'agent de conteneur et chaque tâche en cours d'exécution sur l'instance de conteneur.

Les événements de changement d'état de conteneur et de changement d'état de tâche contiennent deux champs `version` : un dans le corps principal de l'événement et l'autre dans l'objet `detail` de l'événement. Les différences entre ces deux champs sont décrites ci-dessous :

- Le champ `version` du corps de l'événement est défini sur `0` pour tous les événements. Pour plus d'informations sur EventBridge les paramètres, consultez les métadonnées des événements de AWS service dans le guide de EventBridge l'utilisateur Amazon.
- Le champ `version` dans l'objet `detail` de l'événement décrit la version de la ressource associée. Chaque fois qu'une ressource change d'état, cette version est incrémentée. Comme les événements peuvent être envoyés plusieurs fois, ce champ vous permet d'identifier les événements en double. Les événements en double présentent la même version dans l'objet `detail`. Si vous répliquez votre instance de conteneur Amazon ECS et l'état de votre tâche avec EventBridge, vous pouvez comparer la version d'une ressource signalée par Amazon ECS APIs avec la version signalée EventBridge pour la ressource (à l'intérieur de l'`detail`objet) afin de vérifier que la version de votre flux d'événements est à jour.

Les événements d'action de service contiennent uniquement le champ `version` dans le corps principal.

Les événements d'action de service spécifient le service dans deux champs différents :

- Pour les événements générés par`create-service`, le service est `serviceName` sur le terrain.
- Pour les événements générés par`update-service`, le service est `service` sur le terrain.

Si vous utilisez des outils automatisés pour les événements de service, vous devez coder pour les deux champs.

Pour plus d'informations sur la création d'une règle pour les événements liés aux actions de service, consultezÉvénements liés aux actions du service Amazon ECS.

Pour plus d'informations sur la manière d'intégrer Amazon ECS EventBridge, consultez la section Intégration d'Amazon EventBridge et d'Amazon ECS.

Événements de modification de l'état de l'instance de conteneur Amazon ECS

Les scénarios suivants provoquent des événements de changement d'état d'instance de conteneur :

Vous appelez les opérations `StartTaskRunTask`, ou `StopTask` API, directement ou avec le AWS Management Console ou SDKs.

> Le fait de placer ou d'arrêter des tâches sur une instance de conteneur a pour effet de modifier les ressources disponibles sur l'instance de conteneur, comme l'UC, la mémoire et les ports disponibles.

Le planificateur de service Amazon ECS service démarre ou arrête une tâche.

> Le fait de placer ou d'arrêter des tâches sur une instance de conteneur a pour effet de modifier les ressources disponibles sur l'instance de conteneur, comme l'UC, la mémoire et les ports disponibles.

L'agent de conteneur Amazon ECS appelle l'opération d'API `SubmitTaskStateChange` avec un état `STOPPED` pour une tâche ayant l'état souhaité `RUNNING`.

> L'agent de conteneur Amazon ECS supervise l'état des tâches sur vos instances de conteneur, et il signale tout changement d'état. Si une tâche supposée être à l'état `RUNNING` passe à l'état `STOPPED`, l'agent libère les ressources qui ont été allouées à la tâche arrêtée, comme l'UC, la mémoire et les ports disponibles.

Vous désenregistrez l'instance de conteneur à l'aide de l'opération `DeregisterContainerInstance` API, soit directement, soit avec le AWS Management Console ou. SDKs

L'annulation de l'enregistrement d'une instance de conteneur modifie l'état de l'instance de conteneur et l'état de connexion de l'agent de conteneur Amazon ECS.

Une tâche a été arrêtée lorsqu'une EC2 instance a été arrêtée.

Lorsque vous arrêtez une instance de conteneur, les tâches qui sont en cours d'exécution sur elle passe à l'état `STOPPED`.

L'agent de conteneur Amazon ECS enregistre une instance de conteneur pour la première fois.

La première fois que l'agent de conteneur Amazon ECS enregistre une instance de conteneur (lors du lancement ou de la première exécution manuelle), cela crée un événement de changement d'état pour l'instance.

L'agent de conteneur Amazon ECS se connecte à Amazon ECS ou s'en déconnecte.

Lorsque l'agent de conteneur Amazon ECS se connecte ou se déconnecte du backend Amazon ECS, il modifie l'état `agentConnected` de l'instance de conteneur.

> **Note**
>
> L'agent de conteneur Amazon ECS se déconnecte et se reconnecte plusieurs fois par heure dans le cadre de son fonctionnement normal, si bien que des événements de connexion de l'agent sont prévisibles. Ces événements ne signifient pas qu'il existe un problème au niveau de l'agent de conteneur ou de votre instance de conteneur.

Vous mettez à niveau l'agent de conteneur Amazon ECS sur une instance.

Le détail de l'instance de conteneur contient un objet pour la version de l'agent de conteneur. Si vous mettez à niveau l'agent, les informations de cette version changent et génèrent un événement.

Example Événement de changement d'état d'instance de conteneur

Les événements de changement d'état d'instance de conteneur sont remis dans le format suivant. La `detail` section ci-dessous ressemble à l'ContainerInstanceobjet renvoyé par une opération

d'DescribeContainerInstancesAPI dans le manuel Amazon Elastic Container Service API Reference. Pour plus d'informations sur EventBridge les paramètres, consultez les métadonnées des événements de AWS service dans le guide de EventBridge l'utilisateur Amazon.

```
{
  "version": "0",
  "id": "8952ba83-7be2-4ab5-9c32-6687532d15a2",
  "detail-type": "ECS Container Instance State Change",
  "source": "aws.ecs",
  "account": "111122223333",
  "time": "2016-12-06T16:41:06Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ecs:us-east-1:111122223333:container-instance/
b54a2a04-046f-4331-9d74-3f6d7f6ca315"
  ],
  "detail": {
    "agentConnected": true,
    "attributes": [
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
      },
      {
        "name": "com.amazonaws.ecs.capability.task-iam-role-network-host"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.awslogs"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
      },
      {
        "name": "com.amazonaws.ecs.capability.privileged-container"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
      },
```

```
      {
        "name": "com.amazonaws.ecs.capability.ecr-auth"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.20"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.21"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.22"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.23"
      },
      {
        "name": "com.amazonaws.ecs.capability.task-iam-role"
      }
    ],
    "clusterArn": "arn:aws:ecs:us-east-1:111122223333:cluster/default",
    "containerInstanceArn": "arn:aws:ecs:us-east-1:111122223333:container-instance/
b54a2a04-046f-4331-9d74-3f6d7f6ca315",
    "ec2InstanceId": "i-f3a8506b",
    "registeredResources": [
      {
        "name": "CPU",
        "type": "INTEGER",
        "integerValue": 2048
      },
      {
        "name": "MEMORY",
        "type": "INTEGER",
        "integerValue": 3767
      },
      {
        "name": "PORTS",
        "type": "STRINGSET",
        "stringSetValue": [
          "22",
          "2376",
          "2375",
          "51678",
          "51679"
        ]
```

```
      },
      {
        "name": "PORTS_UDP",
        "type": "STRINGSET",
        "stringSetValue": []
      }
    ],
    "remainingResources": [
      {
        "name": "CPU",
        "type": "INTEGER",
        "integerValue": 1988
      },
      {
        "name": "MEMORY",
        "type": "INTEGER",
        "integerValue": 767
      },
      {
        "name": "PORTS",
        "type": "STRINGSET",
        "stringSetValue": [
          "22",
          "2376",
          "2375",
          "51678",
          "51679"
        ]
      },
      {
        "name": "PORTS_UDP",
        "type": "STRINGSET",
        "stringSetValue": []
      }
    ],
    "status": "ACTIVE",
    "version": 14801,
    "versionInfo": {
      "agentHash": "aebcbca",
      "agentVersion": "1.13.0",
      "dockerVersion": "DockerVersion: 1.11.2"
    },
    "updatedAt": "2016-12-06T16:41:06.991Z"
  }
```

```
}
```

Événements de modification de l'état des tâches Amazon ECS

Les scénarios suivants provoquent des événements de changement d'état de tâche :

Vous appelez les opérations `StartTaskRunTask`, ou `StopTask` API, directement ou avec le AWS Management Console AWS CLI, ou SDKs.

Le démarrage ou l'arrêt de tâches crée de nouvelles ressources de tâche ou modifie l'état des ressources de tâche existantes.

Le planificateur de service Amazon ECS service démarre ou arrête une tâche.

Le démarrage ou l'arrêt de tâches crée de nouvelles ressources de tâche ou modifie l'état des ressources de tâche existantes.

L'agent de conteneur Amazon ECS appelle l'opération d'API `SubmitTaskStateChange`.

Pour le type de EC2 lancement Amazon, l'agent de conteneur Amazon ECS surveille l'état de vos tâches sur vos instances de conteneur. L'agent de conteneur Amazon ECS signale tout changement d'état. L'état peut notamment passer de `PENDING` à `RUNNING` ou de `RUNNING` à `STOPPED`.

Vous forcez le désenregistrement de l'instance de conteneur sous-jacente à l'aide de l'opération `DeregisterContainerInstance` API et de l'`force`indicateur, soit directement, soit avec le ou. AWS Management Console SDKs

L'annulation de l'enregistrement d'une instance de conteneur modifie l'état de l'instance de conteneur et l'état de connexion de l'agent de conteneur Amazon ECS. Si les tâches sont en cours d'exécution sur l'instance de conteneur, l'indicateur `force` doit être défini pour permettre l'annulation de l'enregistrement. Cette action arrête toutes les tâches sur l'instance.

L'instance de conteneur sous-jacente est arrêtée ou résiliée.

Lorsque vous arrêtez ou résiliez une instance de conteneur, les tâches qui sont en cours d'exécution sur cette dernière passe à l'état `STOPPED`.

L'état du conteneur de la tâche est modifié.

L'agent de conteneur Amazon ECS supervise l'état des conteneurs dans les tâches. Par exemple, si un conteneur en cours d'exécution dans une tâche s'arrête, le changement d'état de ce dernier génère un événement.

Une tâche utilisant le fournisseur de capacité Fargate Spot reçoit un avis de résiliation.

Lorsqu'une tâche utilise le fournisseur de capacité `FARGATE_SPOT` et qu'elle est arrêtée en raison d'une interruption Spot, un événement de changement d'état de tâche est généré.

Example Événement de modification de l'état de la tâche

Les événements de changement d'état de tâche sont remis dans le format suivant. Notez ce qui suit à propos des champs :

- L'état de santé de l'événement n'est pas disponible dans l'événement de changement d'état de la tâche. Si vous avez besoin de l'état de santé des tâches, vous pouvez exécuter describe-tasks.
- Lorsque vos conteneurs utilisent une image hébergée par Amazon ECR, le `imageDigest` champ est renvoyé.
- Les valeurs des `updatedAt` champs`createdAt`,,`connectivityAt`, `pullStartedAt` `startedAt`pull`StoppedAt`, et sont des horodatages de chaînes ISO.
- La `detail-type` valeur est « Changement d'état de tâche ECS ».
- Lorsque l'événement est généré pour une tâche arrêtée, les `stoppedCode` champs `stoppedReason` et fournissent des informations supplémentaires sur la raison pour laquelle la tâche s'est arrêtée (par exemple, « Initiée par l'utilisateur »).

Pour plus d'informations sur EventBridge les paramètres, consultez les métadonnées des événements de AWS service dans Amazon EventBridge Events Reference.

Pour plus d'informations sur la configuration d'une règle d' EventBridge événement Amazon qui capture uniquement les événements de tâche lorsque l'exécution de la tâche a été interrompue en raison de la fermeture de l'un de ses conteneurs essentiels, voir Envoi d'alertes Amazon Simple Notification Service pour les événements liés à l'arrêt d'une tâche Amazon ECS

```
{
    "version": "0",
    "id": "105f6bb1-4da6-c630-4965-35383018cbca",
    "detail-type": "ECS Task State Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2025-05-06T11:02:34Z",
    "region": "us-east-1",
    "resources": [
```

```
        "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/a1173316d40a45dea9"
    ],
    "detail": {
        "attachments": [
            {
                "id": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                "type": "eni",
                "status": "ATTACHED",
                "details": [
                    {
                        "name": "subnetId",
                        "value": "subnet-0d0eab1bb38d5ca64"
                    },
                    {
                        "name": "networkInterfaceId",
                        "value": "eni-0103a2f01bad57d71"
                    },
                    {
                        "name": "macAddress",
                        "value": "0e:50:d1:c1:77:81"
                    },
                    {
                        "name": "privateDnsName",
                        "value": "ip-10-0-1-163.ec2.internal"
                    },
                    {
                        "name": "privateIPv4Address",
                        "value": "10.0.1.163"
                    }
                ]
            }
        ],
        "attributes": [
            {
                "name": "ecs.cpu-architecture",
                "value": "x86_64"
            }
        ],
        "availabilityZone": "us-east-1b",
        "capacityProviderName": "FARGATE",
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/example-cluster",
        "connectivity": "CONNECTED",
        "connectivityAt": "2025-05-06T11:02:17.19Z",
        "containers": [
```

```
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example-
cluster/a1173316d40a45dea9/a0a99b87-baa8-4bf6-b9f1-a9a95917a635",
                "lastStatus": "RUNNING",
                "name": "web",
                "image": "nginx",
                "imageDigest":
 "sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66",
                "runtimeId": "a1173316d40a45dea9-0265927825",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
                "networkInterfaces": [
                    {
                        "attachmentId": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                        "privateIpv4Address": "10.0.1.163"
                    }
                ],
                "cpu": "99",
                "memory": "100"
            },
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example-
cluster/a1173316d40a45dea9/a2010e2d-ba7c-4135-8b79-e0290ff3cd8c",
                "lastStatus": "RUNNING",
                "name": "aws-guardduty-agent-nm40lC",
                "imageDigest":
 "sha256:bf9197abdf853607e5fa392b4f97ccdd6ca56dd179be3ce8849e552d96582ac8",
                "runtimeId": "a1173316d40a45dea9-2098416933",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
                "networkInterfaces": [
                    {
                        "attachmentId": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                        "privateIpv4Address": "10.0.1.163"
                    }
                ],
                "cpu": "null"
            },
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example-
cluster/a1173316d40a45dea9/dccf0ca2-d929-471f-a5c3-98006fd4379e",
                "lastStatus": "RUNNING",
                "name": "aws-otel-collector",
                "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.32.0",
```

```
                "imageDigest":
 "sha256:7a1b3560655071bcacd66902c20ebe9a69470d5691fe3bd36baace7c2f3c4640",
                "runtimeId": "a1173316d40a45dea9-4027662657",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
                "networkInterfaces": [
                    {
                        "attachmentId": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                        "privateIpv4Address": "10.0.1.163"
                    }
                ],
                "cpu": "0"
            }
        ],
        "cpu": "256",
        "createdAt": "2025-05-06T11:02:13.877Z",
        "desiredStatus": "RUNNING",
        "enableExecuteCommand": false,
        "ephemeralStorage": {
            "sizeInGiB": 20
        },
        "group": "family:webserver",
        "launchType": "FARGATE",
        "lastStatus": "RUNNING",
        "memory": "512",
        "overrides": {
            "containerOverrides": [
                {
                    "name": "web"
                },
                {
                    "environment": [
                        {
                            "name": "CLUSTER_NAME",
                            "value": "example-cluster"
                        },
                        {
                            "name": "REGION",
                            "value": "us-east-1"
                        },
                        {
                            "name": "HOST_PROC",
                            "value": "/host_proc"
                        },
```

```
                        {
                            "name": "AGENT_RUNTIME_ENVIRONMENT",
                            "value": "ecsfargate"
                        },
                        {
                            "name": "STAGE",
                            "value": "prod"
                        }
                    ],
                    "memory": 128,
                    "name": "aws-guardduty-agent-nm40lC"
                },
                {
                    "name": "aws-otel-collector"
                }
            ]
        },
        "platformVersion": "1.4.0",
        "pullStartedAt": "2025-05-06T11:02:24.162Z",
        "pullStoppedAt": "2025-05-06T11:02:33.493Z",
        "startedAt": "2025-05-06T11:02:34.325Z",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/
webserver:5",
        "updatedAt": "2025-05-06T11:02:34.325Z",
        "version": 3
    }
}
```

Example

Voici un exemple d'événement de changement d'état de tâche pour le type de EC2 lancement.

```
{
    "version": "0",
    "id": "a65cf262-f104-0dd5-ceda-4b09ba71a441",
    "detail-type": "ECS Task State Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2025-05-12T13:12:06Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ecs:us-east-1:123456789012:task/example/c1ffa94f19a540ed8d9f7e1d2a5d"
```

```
    ],
    "detail": {
        "attachments": [
            {
                "id": "52333e3b-b812-41a8-b057-9ed184bbe5e1",
                "type": "eni",
                "status": "ATTACHED",
                "details": [
                    {
                        "name": "subnetId",
                        "value": "subnet-0d0eab1bb38d5ca64"
                    },
                    {
                        "name": "networkInterfaceId",
                        "value": "eni-0ea90f746500773a4"
                    },
                    {
                        "name": "macAddress",
                        "value": "0e:d5:9b:ce:49:fb"
                    },
                    {
                        "name": "privateDnsName",
                        "value": "ip-10-0-1-37.ec2.internal"
                    },
                    {
                        "name": "privateIPv4Address",
                        "value": "10.0.1.37"
                    }
                ]
            }
        ],
        "attributes": [
            {
                "name": "ecs.cpu-architecture",
                "value": "x86_64"
            }
        ],
        "availabilityZone": "us-east-1b",
        "capacityProviderName": "Infra-ECS-Cluster-example-fa84e0cc-
AsgCapacityProvider-OseQJU9pizmp",
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/example",
        "connectivity": "CONNECTED",
        "connectivityAt": "2025-05-12T13:11:44.98Z",
```

```
        "containerInstanceArn": "arn:aws:ecs:us-east-1:123456789012:container-instance/
example/d1d84798400f49f3b21cb61610c1e",
        "containers": [
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example/
c1ffa94f19a540ed8d9f7e1d2a5d3626/197d0994-5367-4a6d-9f9a-f075e4a6",
                "lastStatus": "RUNNING",
                "name": "aws-otel-collector",
                "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.32.0",
                "imageDigest":
 "sha256:7a1b3560655071bcacd66902c20ebe9a69470d5691fe3bd36baace7c2f3c4640",
                "runtimeId":
 "8e926f0ccd8fe2b459926f49584ba6d33a3d9f61398dbabe944ee6a13a8ff3a1",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example/
c1ffa94f19a540ed8d9f7e1d2a5d",
                "networkInterfaces": [
                    {
                        "attachmentId": "52333e3b-b812-41a8-b057-9ed184bbe5e1",
                        "privateIpv4Address": "10.0.1.37"
                    }
                ],
                "cpu": "0"
            },
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example/
c1ffa94f19a540ed8d9f7e1d2a5d3626/cab39ef0-9c50-459d-844b-b9d51d73d",
                "lastStatus": "RUNNING",
                "name": "web",
                "image": "nginx",
                "imageDigest":
 "sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66",
                "runtimeId":
 "9f1c73f0094f051541d9e5c2ab1e172d83c4eb5171bcc857c4504b02770ff3b8",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example/
c1ffa94f19a540ed8d9f7e1d2a5d",
                "networkInterfaces": [
                    {
                        "attachmentId": "52333e3b-b812-41a8-b057-9ed184bbe5e1",
                        "privateIpv4Address": "10.0.1.37"
                    }
                ],
                "cpu": "99",
                "memory": "100"
            }
```

```
        ],
        "cpu": "256",
        "createdAt": "2025-05-12T13:11:44.98Z",
        "desiredStatus": "RUNNING",
        "enableExecuteCommand": false,
        "group": "family:webserver",
        "launchType": "EC2",
        "lastStatus": "RUNNING",
        "memory": "512",
        "overrides": {
            "containerOverrides": [
                {
                    "name": "aws-otel-collector"
                },
                {
                    "name": "web"
                }
            ]
        },
        "pullStartedAt": "2025-05-12T13:11:59.491Z",
        "pullStoppedAt": "2025-05-12T13:12:05.896Z",
        "startedAt": "2025-05-12T13:12:06.053Z",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example/
c1ffa94f19a540ed8d9f7e1d2a5d",
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/
webserver",
        "updatedAt": "2025-05-12T13:12:06.053Z",
        "version": 4
    }
}
```

Événements liés aux actions du service Amazon ECS

Amazon ECS envoie des événements d'action de service avec le type de détail Action de service ECS. Contrairement aux événements de changement d'état d'instance de conteneur et de tâche, les événements d'action de service n'incluent pas de numéro de version dans le champ de réponse `details`. Voici un modèle d'événement utilisé pour créer une EventBridge règle pour les événements d'action du service Amazon ECS. Pour plus d'informations, consultez Getting started with EventBridge dans le guide de EventBridge l'utilisateur Amazon.

```
{
```

```
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Service Action"
    ]
}
```

Amazon ECS envoie des événements avec les types d'événements `INFO`, `WARN` et `ERROR`. Les événements d'action de service sont les suivants.

Événements d'action de service avec le type d'événement **`INFO`**

`SERVICE_STEADY_STATE`

Le service est sain et avec le nombre de tâches souhaité, atteignant ainsi un état stable. Le planificateur de service rapporte l'état de façon régulière, vous pouvez donc recevoir ce message plusieurs fois.

`TASKSET_STEADY_STATE`

L'ensemble de tâches est sain et avec le nombre de tâches souhaité, atteignant ainsi un état stable.

`CAPACITY_PROVIDER_STEADY_STATE`

Un fournisseur de capacité associé à un service atteint un état stable.

`SERVICE_DESIRED_COUNT_UPDATED`

Lorsque le planificateur de service met à jour le nombre souhaité calculé pour un service ou un ensemble de tâches. Cet événement n'est pas envoyé lorsque le nombre souhaité est mis à jour manuellement par un utilisateur.

`TASKS_STOPPED`

Le service a arrêté la tâche en cours d'exécution.

`SERVICE_DEPLOYMENT_IN_PROGRESS`

Le déploiement d'un service est en cours. Le déploiement du service peut être une annulation ou une nouvelle révision du service.

`SERVICE_DEPLOYMENT_COMPLETED`

Le déploiement d'un service est stable et est terminé. Le déploiement du service peut prendre la forme d'une annulation ou d'une mise à jour du service.

Événements d'action de service avec le type d'événement **`WARN`**

`SERVICE_TASK_START_IMPAIRED`

Le service n'est pas en mesure de démarrer invariablement les tâches avec succès.

`SERVICE_DISCOVERY_INSTANCE_UNHEALTHY`

Un service utilisant la découverte de service contient une tâche non saine. Le planificateur de service détecte qu'une tâche au sein d'un registre de service n'est pas saine.

`VPC_LATTICE_TARGET_UNHEALTHY`

Le service utilisant VPC Lattice a détecté que l'une des cibles du VPC Lattice est défectueuse.

Événements d'action de service avec le type d'événement **`ERROR`**

`SERVICE_DAEMON_PLACEMENT_CONSTRAINT_VIOLATED`

Une tâche dans un service utilisant la stratégie du planificateur de service `DAEMON` n'est plus conforme à la stratégie de contrainte de placement du service.

`ECS_OPERATION_THROTTLED`

Le planificateur de service a été limité en raison des restrictions de l'API Amazon ECS.

`SERVICE_DISCOVERY_OPERATION_THROTTLED`

Le planificateur de services a été limité en raison des limites de limitation de l' AWS Cloud Map API. Cela peut se produire au niveau des services configurés pour utiliser la découverte de service.

`SERVICE_TASK_PLACEMENT_FAILURE`

Le planificateur de service ne parvient pas à placer une tâche. La cause est décrite dans le champ `reason`.

Une cause courante de la génération de cet événement de service est une quantité insuffisante de ressources dans le cluster pour pouvoir placer la tâche. Par exemple, la capacité de l'UC ou

de la mémoire peut être insuffisante dans les instances de conteneur disponibles, ou aucune instance de conteneur n'est disponible. Une autre cause fréquente est la déconnexion de l'agent de conteneur Amazon ECS dans l'instance de conteneur, ce qui empêche le planificateur de placer la tâche.

`SERVICE_TASK_CONFIGURATION_FAILURE`

Le planificateur de service ne parvient pas à placer une tâche en raison d'une erreur de configuration. La cause est décrite dans le champ `reason`.

L'une des causes courantes de la génération de cet événement de service est l'application de balises au service alors que l'utilisateur ou le rôle n'a pas activé le nouveau format Amazon Resource Name (ARN) dans la région. Pour de plus amples informations, veuillez consulter Amazon Resource Names (ARNs) et IDs. Une autre cause fréquente est l'incapacité d'Amazon ECS à assumer le rôle IAM fourni pour la tâche.

`SERVICE_HEALTH_UNKNOWN`

Le service n'a pas été en mesure de décrire les données de santé relatives aux tâches.

`SERVICE_DEPLOYMENT_FAILED`

Le déploiement d'un service n'a pas atteint son paroxysme. Cela se produit lorsqu'un CloudWatch est déclenché ou que le disjoncteur détecte un échec du déploiement du service.

Example Événement d'état stable de service

Les événements d'état stable de service sont remis dans le format suivant. Pour plus d'informations sur EventBridge les paramètres, consultez la section Events EventBridge in du guide de EventBridge l'utilisateur Amazon.

```
{
    "version": "0",
    "id": "af3c496d-f4a8-65d1-70f4-a69d52e9b584",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:27:22Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
```

```
        "eventType": "INFO",
        "eventName": "SERVICE_STEADY_STATE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "createdAt": "2019-11-19T19:27:22.695Z"
    }
}
```

Example Événement d'état stable de fournisseur de capacité

Les événements d'état stable de fournisseur de capacité sont remis dans le format suivant.

```
{
    "version": "0",
    "id": "b9baa007-2f33-0eb1-5760-0d02a572d81f",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:37:00Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "INFO",
        "eventName": "CAPACITY_PROVIDER_STEADY_STATE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "capacityProviderArns": [
            "arn:aws:ecs:us-west-2:111122223333:capacity-provider/ASG-tutorial-
capacity-provider"
        ],
        "createdAt": "2019-11-19T19:37:00.807Z"
    }
}
```

Example Événement de lancement de tâche de service défaillant

Les événements de lancement de tâche de service défaillant sont remis dans le format suivant.

```
{
    "version": "0",
    "id": "57c9506e-9d21-294c-d2fe-e8738da7e67d",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
```

```
    "account": "111122223333",
    "time": "2019-11-19T19:55:38Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "WARN",
        "eventName": "SERVICE_TASK_START_IMPAIRED",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "createdAt": "2019-11-19T19:55:38.725Z"
    }
}
```

Example Événement d'échec de placement de tâche de service

Les événements d'échec de placement de tâche de service sont remis dans le format suivant. Pour plus d'informations, consultez la section Événements EventBridge du guide de EventBridge l'utilisateur Amazon.

Dans l'exemple suivant, la tâche tentait d'utiliser le fournisseur de capacité `FARGATE_SPOT`, mais le planificateur de service n'a pas pu acquérir de capacité Fargate Spot.

```
{
    "version": "0",
    "id": "ddca6449-b258-46c0-8653-e0e3a6d0468b",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:55:38Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "ERROR",
        "eventName": "SERVICE_TASK_PLACEMENT_FAILURE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "capacityProviderArns": [
            "arn:aws:ecs:us-west-2:111122223333:capacity-provider/FARGATE_SPOT"
        ],
        "reason": "RESOURCE:FARGATE",
        "createdAt": "2019-11-06T19:09:33.087Z"
```

```
    }
}
```

Dans l'exemple suivant concernant le type de EC2 lancement, la tâche a été tentée de lancer sur l'instance de conteneur, `2dd1b186f39845a584488d2ef155c131` mais le planificateur de services n'a pas pu la placer en raison d'un processeur insuffisant.

```
{
  "version": "0",
  "id": "ddca6449-b258-46c0-8653-e0e3a6d0468b",
  "detail-type": "ECS Service Action",
  "source": "aws.ecs",
  "account": "111122223333",
  "time": "2019-11-19T19:55:38Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
  ],
  "detail": {
    "eventType": "ERROR",
    "eventName": "SERVICE_TASK_PLACEMENT_FAILURE",
    "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
    "containerInstanceArns": [
    "arn:aws:ecs:us-west-2:111122223333:container-instance/
default/2dd1b186f39845a584488d2ef155c131"
    ],
    "reason": "RESOURCE:CPU",
    "createdAt": "2019-11-06T19:09:33.087Z"
  }
}
```

Événements de changement d'état du déploiement du service Amazon ECS

Amazon ECS envoie des événements de modification d'état de déploiement de service avec le type de détail ECS Deployment State Change (Modification d'état de déploiement ECS). Voici un modèle d'événement utilisé pour créer une EventBridge règle pour les événements de changement d'état du déploiement du service Amazon ECS. Pour plus d'informations sur la création d'une EventBridge règle, consultez Getting started with Amazon EventBridge dans le guide de EventBridge l'utilisateur Amazon.

```
{
```

```
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Deployment State Change"
    ]
}
```

Amazon ECS envoie des événements avec les types d'événements `INFO` et `ERROR`. Pour de plus amples informations, consultez Événements liés aux actions du service Amazon ECS.

Vous trouverez ci-dessous des événements de modification de l'état de déploiement de service.

`SERVICE_DEPLOYMENT_IN_PROGRESS`

Le déploiement du service est en cours. Cet événement est envoyé pour les déploiements initiaux et les déploiements de restauration.

`SERVICE_DEPLOYMENT_COMPLETED`

Le déploiement du service est terminé. Cet événement est envoyé lorsqu'un service atteint un état stable après un déploiement.

`SERVICE_DEPLOYMENT_FAILED`

Le déploiement du service a échoué. Cet événement est envoyé pour les services avec la logique de disjoncteur de circuit de déploiement activée.

Example déploiement de service dans l'événement de progression

Le déploiement de service dans les événements de progression est fourni lors du démarrage d'un déploiement initial et de restauration. La différence entre les deux est dans le champ `reason`. Pour plus d'informations sur EventBridge les paramètres, consultez les métadonnées des événements de AWS service dans le guide de EventBridge l'utilisateur Amazon.

Vous trouverez ci-dessous un exemple de sortie pour le démarrage d'un déploiement initial.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3a6EXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
```

```
    "time": "2020-05-23T12:31:14Z",
    "region": "us-west-2",
    "resources": [
         "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
         "eventType": "INFO",
         "eventName": "SERVICE_DEPLOYMENT_IN_PROGRESS",
         "deploymentId": "ecs-svc/123",
         "updatedAt": "2020-05-23T11:11:11Z",
         "reason": "ECS deployment deploymentId in progress."
    }
}
```

Vous trouverez ci-dessous un exemple de sortie pour le démarrage d'un déploiement de restauration. Le champ `reason` fournit l'ID du déploiement vers lequel le service est en cours de restauration.

```
{
    "version": "0",
    "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
    "detail-type": "ECS Deployment State Change",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2020-05-23T12:31:14Z",
    "region": "us-west-2",
    "resources": [
         "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
         "eventType": "INFO",
         "eventName": "SERVICE_DEPLOYMENT_IN_PROGRESS",
         "deploymentId": "ecs-svc/123",
         "updatedAt": "2020-05-23T11:11:11Z",
         "reason": "ECS deployment circuit breaker: rolling back to
 deploymentId deploymentID."
    }
}
```

Example événement de déploiement de service terminé

Les événements à l'état terminé de déploiement de service sont remis dans le format suivant. Pour de plus amples informations, veuillez consulter Déployez les services Amazon ECS en remplaçant les tâches.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
   "region": "us-west-2",
   "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
   ],
   "detail": {
        "eventType": "INFO",
        "eventName": "SERVICE_DEPLOYMENT_COMPLETED",
        "deploymentId": "ecs-svc/123",
        "updatedAt": "2020-05-23T11:11:11Z",
        "reason": "ECS deployment deploymentID completed."
   }
}
```

Example événement échoué de déploiement de service

Les événements à l'état échoué de déploiement de service sont remis dans le format suivant. Un événement à l'état échoué de déploiement de service n'est envoyé que pour les services dont la logique de disjoncteur de circuit de déploiement est activée. Pour de plus amples informations, veuillez consulter Déployez les services Amazon ECS en remplaçant les tâches.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
   "region": "us-west-2",
   "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
   ],
   "detail": {
        "eventType": "ERROR",
        "eventName": "SERVICE_DEPLOYMENT_FAILED",
        "deploymentId": "ecs-svc/123",
```

```
        "updatedAt": "2020-05-23T11:11:11Z",
        "reason": "ECS deployment circuit breaker: task failed to start."
    }
}
```

Gestion des événements Amazon ECS

Amazon ECS envoie des événements au moins une fois. Cela signifie que vous pouvez recevoir plusieurs copies d'un événement donné. En outre, des événements peuvent ne pas être transmis aux écouteurs d'événements dans l'ordre dans lequel ils se produisent.

Pour bien ordonner les événements, la section `detail` de chaque événement contient une propriété `version`. Chaque fois qu'une ressource change d'état, cette `version` est incrémentée. Les événements en double présentent la même `version` dans l'objet `detail`. Si vous répliquez votre instance de conteneur Amazon ECS et l'état de votre tâche avec EventBridge, vous pouvez comparer la version d'une ressource signalée par Amazon ECS APIs avec celle `version` signalée EventBridge pour la ressource afin de vérifier que la version de votre flux d'événements est à jour. Les événements avec un numéro de propriétés de version plus élevé doivent être traités comme postérieurs aux événements avec des numéros de version inférieurs.

Exemple : gestion des événements dans une fonction AWS Lambda

L'exemple suivant montre une fonction Lambda écrite en Python 3.9 qui capture les événements de changement d'état de tâche et d'instance de conteneur et les enregistre dans l'une des deux tables Amazon DynamoDB :

- ECSCtrInstanceState— Stocke le dernier état d'une instance de conteneur. L'ID de table correspond à la valeur `containerInstanceArn` de l'instance de conteneur.
- ECSTaskÉtat : enregistre le dernier état d'une tâche. L'ID de table correspond à la valeur `taskArn` de la tâche.

```
import json
import boto3

def lambda_handler(event, context):
    id_name = ""
    new_record = {}

    # For debugging so you can see raw event format.
```

```
    print('Here is the event:')
    print((json.dumps(event)))

    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source type
 of: aws.ecs")

    # Switch on task/container events.
    table_name = ""
    if event["detail-type"] == "ECS Task State Change":
        table_name = "ECSTaskState"
        id_name = "taskArn"
        event_id = event["detail"]["taskArn"]
    elif event["detail-type"] == "ECS Container Instance State Change":
        table_name = "ECSCtrInstanceState"
        id_name =  "containerInstanceArn"
        event_id = event["detail"]["containerInstanceArn"]
    else:
        raise ValueError("detail-type for event is not a supported type. Exiting
 without saving event.")

    new_record["cw_version"] = event["version"]
    new_record.update(event["detail"])

    # "status" is a reserved word in DDB, but it appears in containerPort
    # state change messages.
    if "status" in event:
        new_record["current_status"] = event["status"]
        new_record.pop("status")


    # Look first to see if you have received a newer version of an event ID.
    # If the version is OLDER than what you have on file, do not process it.
    # Otherwise, update the associated record with this latest information.
    print("Looking for recent event with same ID...")
    dynamodb = boto3.resource("dynamodb", region_name="us-east-1")
    table = dynamodb.Table(table_name)
    saved_event = table.get_item(
        Key={
            id_name : event_id
        }
    )
    if "Item" in saved_event:
        # Compare events and reconcile.
```

```
        print(("EXISTING EVENT DETECTED: Id " + event_id + " - reconciling"))
        if saved_event["Item"]["version"] < event["detail"]["version"]:
            print("Received event is a more recent version than the stored event -
 updating")
            table.put_item(
                Item=new_record
            )
        else:
            print("Received event is an older version than the stored event -
 ignoring")
    else:
        print(("Saving new event - ID " + event_id))

        table.put_item(
            Item=new_record
        )
```

L'exemple de Fargate suivant montre une fonction Lambda écrite en Python 3.9 qui capture les événements de changement d'état des tâches et les enregistre dans la table Amazon DynamoDB suivante :

```
import json
import boto3

def lambda_handler(event, context):
    id_name = ""
    new_record = {}

    # For debugging so you can see raw event format.
    print('Here is the event:')
    print((json.dumps(event)))

    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source type
 of: aws.ecs")

    # Switch on task/container events.
    table_name = ""
    if event["detail-type"] == "ECS Task State Change":
        table_name = "ECSTaskState"
        id_name = "taskArn"
        event_id = event["detail"]["taskArn"]
    else:
```

```
        raise ValueError("detail-type for event is not a supported type. Exiting
without saving event.")

    new_record["cw_version"] = event["version"]
    new_record.update(event["detail"])

    # "status" is a reserved word in DDB, but it appears in containerPort
    # state change messages.
    if "status" in event:
        new_record["current_status"] = event["status"]
        new_record.pop("status")


    # Look first to see if you have received a newer version of an event ID.
    # If the version is OLDER than what you have on file, do not process it.
    # Otherwise, update the associated record with this latest information.
    print("Looking for recent event with same ID...")
    dynamodb = boto3.resource("dynamodb", region_name="us-east-1")
    table = dynamodb.Table(table_name)
    saved_event = table.get_item(
        Key={
            id_name : event_id
        }
    )
    if "Item" in saved_event:
        # Compare events and reconcile.
        print(("EXISTING EVENT DETECTED: Id " + event_id + " - reconciling"))
        if saved_event["Item"]["version"] < event["detail"]["version"]:
            print("Received event is a more recent version than the stored event -
updating")
            table.put_item(
                Item=new_record
            )
        else:
            print("Received event is an older version than the stored event -
ignoring")
    else:
        print(("Saving new event - ID " + event_id))

        table.put_item(
            Item=new_record
        )
```

Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée

Container Insights collecte, agrège et résume les métriques et les journaux de vos applications conteneurisées et de vos microservices. Container Insights, avec une observabilité améliorée, fournit tous les indicateurs de Container Insights, ainsi que des indicateurs supplémentaires sur les tâches et les conteneurs.

Container Insights découvre tous les conteneurs en cours d'exécution dans un cluster et collecte des données de performance à chaque couche de la pile de performances. Les données d'exploitation sont collectées en tant qu'événements de journaux de performances. Ces entrées utilisent un schéma JSON structuré qui permet aux données à haute cardinalité d'être intégrées et stockées à grande échelle. À partir de ces données, CloudWatch crée des métriques agrégées de niveau supérieur au niveau du cluster, du service et de la tâche sous forme CloudWatch de métriques. Les métriques incluent l'utilisation des ressources telles que l'UC, la mémoire, le disque et le réseau. Les métriques sont disponibles dans les tableaux de bord CloudWatch automatiques.

Les métriques reflètent uniquement les ressources dont les tâches sont en cours d'exécution pendant la période spécifiée. Par exemple, si vous avez un cluster avec un service mais que ce service n'a aucune tâche avec l'état `RUNNING`, aucune métrique n'est envoyée à CloudWatch. Si vous disposez de deux services et que l'un d'eux seulement a des tâches en cours d'exécution, seules les métriques du service avec des tâches en cours d'exécution seront envoyées.

Le 2 décembre 2024, AWS a publié Container Insights avec une observabilité améliorée pour Amazon ECS. Cette version prend en charge une observabilité améliorée pour les clusters Amazon ECS utilisant les types de lancement Amazon EC2 et Fargate. Une fois que vous avez configuré Container Insights avec une observabilité améliorée sur Amazon ECS, Container Insights collecte automatiquement des données télémétriques détaillées sur l'infrastructure, du niveau du cluster jusqu'au niveau du conteneur dans votre environnement, et affiche ces données de performance critiques dans des tableaux de bord organisés, ce qui élimine les lourdeurs liées à la configuration de l'observabilité. Pour plus d'informations sur la façon de configurer Container Insights avec une observabilité améliorée, consultezContainer Insights avec une meilleure observabilité.

Nous vous recommandons d'utiliser Container Insights avec une observabilité améliorée plutôt que Container Insights, car cela fournit une visibilité détaillée de votre environnement de conteneurs, réduisant ainsi le délai moyen de résolution. Pour plus d'informations, consultez Amazon ECS Container Insights avec des métriques d'observabilité améliorées dans le guide de l' CloudWatch utilisateur Amazon.

Les événements que vous pouvez consulter sont ceux qu'Amazon ECS envoie à Amazon EventBridge. Pour plus d'informations, consultez Événements Amazon ECS.

> **Important**
>
> Les métriques collectées par CloudWatch Container Insights sont facturées en tant que métriques personnalisées. Pour plus d'informations sur la CloudWatch tarification, consultez la section CloudWatchTarification.

Déterminer l'état des tâches Amazon ECS à l'aide de vérifications de l'état des conteneurs

Lorsque vous créez une définition de tâche, vous pouvez configurer un contrôle de santé pour vos conteneurs. Les contrôles de santé sont des commandes qui s'exécutent localement sur un conteneur et valident l'état et la disponibilité de l'application.

L'agent de conteneur Amazon ECS assure une surveillance et génère des rapports uniquement pour les vérifications d'état spécifiées dans la définition de la tâche. Amazon ECS n'assure pas la surveillance de l'état Docker qui est intégrée dans une image de conteneur, mais qui n'est pas spécifiée dans la définition du conteneur. Les paramètres de surveillance de l'état spécifiés dans une définition du conteneur remplacent toutes les surveillances de l'état Docker présentes dans l'image de conteneur.

Lorsqu'un contrôle de santé est défini dans une définition de tâche, le conteneur exécute le processus de contrôle de santé à l'intérieur du conteneur, puis évalue le code de sortie pour déterminer l'état de santé de l'application.

Le bilan de santé comprend les paramètres suivants :

- Commande : commande exécutée par le conteneur pour déterminer s'il est sain. Le tableau de chaînes peut commencer par `CMD` pour exécuter directement les arguments de la commande, ou par `CMD-SHELL` pour exécuter la commande avec le shell par défaut du conteneur. `CMD-SHELL`À utiliser lorsque vous avez besoin de fonctionnalités du shell telles que les canaux, les redirections, le chaînage de commandes ou l'extension de variables d'environnement. Par exemple, vous `CMD-SHELL` permet d'utiliser des commandes telles que `curl -f http://localhost/ || exit 1` celles où le shell interprète l'`||`opérateur. `CMD`À utiliser pour les commandes simples qui ne nécessitent pas d'interprétation du shell.

- Intervalle : intervalle de temps (en secondes) entre chaque bilan de santé.
- Délai d'attente : délai (en secondes) nécessaire pour qu'un bilan de santé réussisse avant qu'il ne soit considéré comme un échec.
- Réessais : nombre de fois où il faut réessayer un contrôle de santé qui a échoué avant que le contenant ne soit considéré comme insalubre.
- Période de début : période de grâce facultative qui donne aux conteneurs le temps de démarrer avant que les tests de santé échoués ne soient pris en compte dans le nombre maximum de tentatives.

  Si une vérification de l'état aboutit dans le délai défini par `startPeriod`, le conteneur est considéré comme sain et toutes les défaillances suivantes sont prises en compte dans le nombre maximal de nouvelles tentatives.

Pour plus d'informations sur la manière de spécifier un bilan de santé dans la définition d'une tâche, consultezSurveillance de l'état.

Ce qui suit décrit les valeurs d'état de santé possibles pour un conteneur :

- `HEALTHY` : le conteneur a passé avec succès la surveillance de l'état.
- `UNHEALTHY` : le conteneur a échoué à la surveillance de l'état.
- `UNKNOWN` : la surveillance de l'état du conteneur est en cours d'évaluation, aucune surveillance de l'état du conteneur n'est définie ou Amazon ECS n'a pas l'état de santé du conteneur.

Les commandes de contrôle de santé s'exécutent sur le conteneur. Vous devez donc inclure les commandes dans l'image du conteneur.

Le bilan de santé se connecte à l'application via l'interface de bouclage du conteneur à `localhost` ou`127.0.0.1`. Un code de sortie égal à `0` indique un succès et un code de sortie différent de zéro indique un échec.

Tenez compte des points suivants lorsque vous utilisez des contrôles de santé des conteneurs :

- Les surveillances de l'état du conteneur requièrent la version 1.17.0 ou une version supérieure de l'agent de conteneur Amazon ECS.
- Les contrôles de santé des conteneurs sont pris en charge pour les tâches Fargate si vous utilisez une version de plate-forme Linux ou supérieure ou une `1.1.0` version de plate-forme Windows ou supérieure `1.1.0`

Comment Amazon ECS détermine l'état de santé des tâches

Les conteneurs essentiels et dotés d'une commande de vérification de l'état dans la définition de la tâche sont les seuls pris en compte pour déterminer l'état de santé de la tâche.

Les règles suivantes sont évaluées dans l'ordre :

1. Si le statut d'un conteneur essentiel est`UNHEALTHY`, le statut de la tâche est`UNHEALTHY`.
2. Si le statut d'un conteneur essentiel est`UNKNOWN`, le statut de la tâche est`UNKNOWN`.
3. Si le statut de tous les conteneurs essentiels est le `HEALTHY` même, le statut de la tâche est`HEALTHY`.

Prenons l'exemple d'état des tâches suivant avec deux conteneurs essentiels.

Conteneur 1 : santé	Container 2 : santé	Santé des tâches
`UNHEALTHY`	`UNKNOWN`	`UNHEALTHY`
`UNHEALTHY`	`HEALTHY`	`UNHEALTHY`
`HEALTHY`	`UNKNOWN`	`UNKNOWN`
`HEALTHY`	`HEALTHY`	`HEALTHY`

Prenons l'exemple d'intégrité des tâches suivant avec 3 conteneurs.

Conteneur 1 : santé	Container 2 : santé	Container 3 : santé	Santé des tâches
`UNHEALTHY`	`UNKNOWN`	`UNKNOWN`	`UNHEALTHY`
`UNHEALTHY`	`UNKNOWN`	`HEALTHY`	`UNHEALTHY`
`UNHEALTHY`	`HEALTHY`	`HEALTHY`	`UNHEALTHY`
`HEALTHY`	`UNKNOWN`	`HEALTHY`	`UNKNOWN`
`HEALTHY`	`UNKNOWN`	`UNKNOWN`	`UNKNOWN`

Conteneur 1 : santé	Container 2 : santé	Container 3 : santé	Santé des tâches
HEALTHY	HEALTHY	HEALTHY	HEALTHY

Comment les bilans de santé sont-ils affectés par les déconnexions d'agents

Si l'agent de conteneur Amazon ECS est déconnecté du service Amazon ECS, le conteneur ne passera pas à un `UNHEALTHY` statut. Cela a été conçu pour garantir que les conteneurs continuent de fonctionner pendant les redémarrages de l'agent ou en cas d'indisponibilité temporaire. L'état du bilan de santé est la réponse « dernière réponse » de l'agent Amazon ECS. Ainsi, si le conteneur a été pris en compte `HEALTHY` avant la déconnexion, cet état restera en vigueur jusqu'à ce que l'agent se reconnecte et qu'un autre contrôle de santé ait lieu. Aucune hypothèse n'a été émise quant au statut des surveillances de l'état des conteneurs.

Affichage de l'état des conteneurs Amazon ECS

Vous pouvez consulter l'état du conteneur dans la console et utiliser l'API dans la `DescribeTasks` réponse. Pour plus d'informations, consultez le DescribeTasksmanuel Amazon Elastic Container Service API Reference.

Si vous utilisez la journalisation pour votre conteneur, par exemple Amazon CloudWatch Logs, vous pouvez configurer la commande de vérification de l'état de santé pour transférer les résultats relatifs à l'état du conteneur vers vos journaux. Assurez-vous de l'utiliser `2&1` pour attraper à la fois les `stderr` informations `stdout` et. L'exemple suivant l'utilise `CMD-SHELL` car il nécessite des fonctionnalités du shell telles que les opérateurs pipe (>>) et logique OR (`||`) :

```
"command": [
     "CMD-SHELL",
     "curl -f http://localhost/ >> /proc/1/fd/1 2>&1  || exit 1"
   ],
```

Surveiller l'état de l'instance de conteneur Amazon ECS

Amazon ECS fournit la surveillance de l'état des instances de conteneur. Vous pouvez rapidement déterminer si Amazon ECS a détecté des problèmes susceptibles d'empêcher vos instances de

conteneur d'exécuter des conteneurs. Amazon ECS effectue des contrôles automatisés sur chaque instance de conteneur en cours d'exécution avec la version d'agent `1.57.0` ou ultérieure pour identifier les problèmes. Pour plus d'informations sur la vérification de la version d'agent et d'une instance de conteneur, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Vous devez utiliser la AWS CLI version `1.22.3` ou une version ultérieure ou une AWS CLI version `2.3.6` ou une version ultérieure. Pour plus d'informations sur la mise à jour du AWS CLI, voir Installation ou mise à jour de la AWS CLI dernière version du Guide de l'AWS Command Line Interface utilisateur, version 2.

Pour afficher l'état de santé de l'instance de conteneur, exécutez `describe-container-instances` l'`CONTAINER_INSTANCE_HEALTH`option.

Voici les valeurs valides pour `overallStatus` :

- `OK`
- `IMPAIRED`
- `INSUFFICIENT_DATA`
- `INITIALIZING`

Voici un exemple d'exécution`describe-container-instances`.

```
aws ecs describe-container-instances \
     --cluster cluster_name \
     --container-instances 47279cd2cadb41cbaef2dcEXAMPLE \
     --include CONTAINER_INSTANCE_HEALTH
```

Voici un exemple de l'objet du statut de l'état dans la sortie.

```
"healthStatus": {
 "overallStatus": "OK",
 "details": [{
  "type": "CONTAINER_RUNTIME",
  "status": "OK",
  "lastUpdated": "2021-11-10T03:30:26+00:00",
  "lastStatusChange": "2021-11-10T03:26:41+00:00"
 }]
}
```

Problèmes liés à l'état de santé des instances de conteneurs

S'il s'`overallStatus`agit d'un statut autre que`OK`, essayez ce qui suit :

- Attendez, puis exécutez `describe-container-instances`
- Consultez l'état de votre instance de conteneur dans la EC2 console ou à l'aide de la CLI.
- Passez en revue les CloudWatch indicateurs. Pour de plus amples informations, consultez Surveillez Amazon ECS à l'aide de CloudWatch.
- Vérifiez le AWS Health Dashboard pour voir s'il y a des problèmes avec le service.

Identifiez les opportunités d'optimisation d'Amazon ECS à l'aide des données de suivi des applications

Amazon ECS s'intègre à AWS Distro pour collecter des données de suivi OpenTelemetry à partir de votre application. Amazon ECS utilise un conteneur AWS Distro for OpenTelemetry sidecar pour collecter et acheminer les données de suivi. AWS X-Ray Pour plus d'informations, consultez Configuration de AWS Distro pour OpenTelemetry Collector dans Amazon ECS. Vous pouvez ensuite l'utiliser AWS X-Ray pour identifier les erreurs et les exceptions, analyser les problèmes de performance et les temps de réponse.

Pour que AWS Distro for OpenTelemetry Collector envoie des données de trace AWS X-Ray, votre application doit être configurée pour créer les données de trace. Pour plus d'informations, consultez Instrumenter votre application pour AWS X-Ray dans le Guide du développeur AWS X-Ray .

Autorisations IAM requises pour AWS Distro pour OpenTelemetry l'intégration avec AWS X-Ray

L'intégration d'Amazon ECS à AWS Distro for OpenTelemetry nécessite que vous créiez un rôle de tâche et que vous le spécifiiez dans la définition de votre tâche. Nous vous recommandons de configurer la AWS distribution pour le OpenTelemetry sidecar afin d'acheminer les journaux des conteneurs vers les journaux. CloudWatch

> **Important**
>
> Si vous collectez également des métriques d'application à l'aide de AWS Distro pour OpenTelemetry l'intégration, assurez-vous que le rôle IAM de votre tâche contient également

les autorisations nécessaires à cette intégration. Pour de plus amples informations, veuillez consulter Corrélez les performances des applications Amazon ECS à l'aide des métriques des applications.

Créez la politique suivante, puis associez-la au rôle d'exécution de la tâche.

Pour utiliser l'éditeur de politique JSON afin de créer une politique

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).

   Si vous sélectionnez Politiques pour la première fois, la page Bienvenue dans les politiques gérées s'affiche. Sélectionnez Mise en route.
3. En haut de la page, sélectionnez Créer une politique.
4. Dans la section Éditeur de politique, choisissez l'option JSON.
5. Entrez le document de politique JSON suivant :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:PutLogEvents",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:DescribeLogGroups",
                "logs:PutRetentionPolicy",
                "xray:PutTraceSegments",
                "xray:PutTelemetryRecords",
                "xray:GetSamplingRules",
                "xray:GetSamplingTargets",
                "xray:GetSamplingStatisticSummaries",
                "ssm:GetParameters"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

6. Choisissez Suivant.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez la page Restructuration de politique dans le Guide de l'utilisateur IAM.

7. Sur la page Vérifier et créer, saisissez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique.
8. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Spécification de la AWS distribution pour le OpenTelemetry sidecar à AWS X-Ray intégrer dans la définition de votre tâche

La console Amazon ECS simplifie la création du conteneur AWS Distro for OpenTelemetry sidecar en utilisant l'option Use trace collection. Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.

Si vous n'utilisez pas la console Amazon ECS, vous pouvez ajouter le conteneur AWS Distro for OpenTelemetry sidecar à votre définition de tâche. L'extrait de définition de tâche suivant montre la définition du conteneur pour l'ajout de AWS Distro pour le OpenTelemetry sidecar à des fins d'intégration. AWS X-Ray

```
{
 "family": "otel-using-xray",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryXrayRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [{
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
```

```
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
    }
   },
   "dependsOn": [{
    "containerName": "aws-otel-collector",
    "condition": "START"
   }]
  },
  {
   "name": "aws-otel-collector",
   "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
   "essential": true,
   "command": [
    "--config=/etc/ecs/otel-instance-metrics-config.yaml"
   ],
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "True",
     "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
    }
   }
  }
 ],
 "networkMode": "awsvpc",
 "requiresCompatibilities": [
  "FARGATE"
 ],
 "cpu": "1024",
 "memory": "3072"
}
```

Corrélez les performances des applications Amazon ECS à l'aide des métriques des applications

Amazon ECS on Fargate permet de collecter des métriques à partir de vos applications exécutées sur Fargate et de les exporter vers Amazon CloudWatch ou Amazon Managed Service for Prometheus.

Vous pouvez utiliser les métadonnées collectées pour corréler les données de performance des applications avec les données d'infrastructure sous-jacentes, réduisant ainsi le temps moyen de résolution du problème.

Amazon ECS utilise un conteneur AWS Distro for OpenTelemetry sidecar pour collecter et acheminer les métriques de votre application vers leur destination. L'expérience de la console Amazon ECS simplifie le processus d'ajout de cette intégration lors de la création de vos définitions de tâches.

Rubriques

- Exporter les métriques des applications vers Amazon CloudWatch
- Exportation des métriques d'application vers Amazon Managed Service for Prometheus

Exporter les métriques des applications vers Amazon CloudWatch

Amazon ECS on Fargate prend en charge l'exportation de vos métriques d'application personnalisées vers CloudWatch Amazon en tant que métriques personnalisées. Cela se fait en ajoutant le conteneur AWS Distro for OpenTelemetry sidecar à votre définition de tâche. La console Amazon ECS simplifie ce processus en ajoutant l'option Utiliser la collecte de métriques lors de la création d'une nouvelle définition de tâche. Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.

Les métriques de l'application sont exportées vers CloudWatch Logs avec le nom du groupe de journaux `/aws/ecs/application/metrics` et les métriques peuvent être consultées dans l'espace de `ECS/AWSOTel/Application` noms. Votre application doit être équipée du OpenTelemetry SDK. Pour plus d'informations, voir Introduction à AWS Distro for OpenTelemetry in the AWS Distro pour OpenTelemetry la documentation.

Considérations

Les points suivants doivent être pris en compte lors de l'utilisation de l'intégration Amazon ECS on Fargate AWS avec Distro OpenTelemetry pour envoyer des métriques d'application à Amazon. CloudWatch

- Cette intégration envoie uniquement les métriques personnalisées de votre application à CloudWatch. Si vous souhaitez obtenir des statistiques au niveau des tâches, vous pouvez activer Container Insights dans la configuration du cluster Amazon ECS. Pour de plus amples informations, veuillez consulter Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée.
- La AWS distribution pour l' OpenTelemetry intégration est prise en charge pour les charges de travail Amazon ECS hébergées sur Fargate et les charges de travail Amazon ECS hébergées sur des instances Amazon. EC2 Les instances externes ne sont actuellement pas prises en charge.
- CloudWatch prend en charge un maximum de 30 dimensions par métrique. Par défaut, Amazon ECS inclut par défaut les dimensions `TaskARN`, `ClusterARN`, `LaunchType`, `TaskDefinitionFamily`, et `TaskDefinitionRevision` aux métriques. Les 25 dimensions restantes peuvent être définies par votre application. Si plus de 30 dimensions sont configurées, CloudWatch impossible de les afficher. Dans ce cas, les métriques de l'application apparaîtront dans l'espace de noms des `ECS/AWSOTel/Application` CloudWatch métriques, mais sans aucune dimension. Vous pouvez instrumenter votre application pour ajouter des dimensions supplémentaires. Pour plus d'informations, consultez la section Utilisation CloudWatch des métriques avec AWS Distro pour OpenTelemetry dans la distribution pour OpenTelemetry obtenir de la AWS documentation.

Autorisations IAM requises pour AWS Distro pour OpenTelemetry l'intégration à Amazon CloudWatch

L'intégration d'Amazon ECS à AWS Distro for OpenTelemetry nécessite que vous créiez un rôle IAM de tâche et que vous le spécifiiez dans la définition de votre tâche. Nous recommandons que la AWS distribution pour OpenTelemetry sidecar soit également configurée pour acheminer les journaux des conteneurs vers les journaux, ce qui nécessite la création et la spécification d'un rôle IAM d'exécution de tâches dans votre définition de tâche. CloudWatch La console Amazon ECS prend en charge le rôle IAM d'exécution des tâches en votre nom, mais le rôle IAM des tâches doit être créé manuellement et ajouté à votre définition de tâche. Pour plus d'informations sur le rôle IAM d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS.

> **Important**
>
> Si vous collectez également des données de suivi d'applications à l'aide de AWS Distro à des fins OpenTelemetry d'intégration, assurez-vous que le rôle IAM de votre tâche contient également les autorisations nécessaires à cette intégration. Pour de plus amples informations, veuillez consulter Identifiez les opportunités d'optimisation d'Amazon ECS à l'aide des données de suivi des applications.
>
> Si votre application nécessite des autorisations supplémentaires, vous devez les ajouter à cette politique. Chaque définition de tâche ne peut spécifier qu'un rôle IAM de tâche. Par exemple, si vous utilisez un fichier de configuration personnalisé stocké dans Systems Manager, vous devez ajouter l'autorisation `ssm:GetParameters` à cette politique IAM.

Pour créer le rôle de service pour Elastic Container Service (console IAM)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez Elastic Container Service, puis choisissez le cas d'utilisation d'Elastic Container Service Task.
5. Choisissez Suivant.
6. Dans la section Ajouter des autorisations, recherchez AWSDistroOpenTelemetryPolicyForXray, puis sélectionnez la politique.
7. (Facultatif) Définissez une limite d'autorisations. Il s’agit d’une fonctionnalité avancée disponible pour les fonctions de service, mais pas pour les rôles liés à un service.

   a. Ouvrez la section Définir une limite des autorisations et choisissez Utiliser une limite des autorisations pour contrôler le nombre maximum d’autorisations de rôle.

      IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte.

   b. Sélectionnez la politique à utiliser comme limite d'autorisations.
8. Choisissez Suivant.
9. Entrez un nom de rôle ou un suffixe de nom de rôle pour vous aider à identifier l'objectif du rôle.

> **Important**
>
> Lorsque vous nommez un rôle, notez ce qui suit :
>
> - Les noms de rôles doivent être uniques au sein du Compte AWS vôtre et ne peuvent pas être rendus uniques au cas par cas.
>
> Par exemple, ne créez pas deux rôles nommés **`PRODROLE`** et **`prodrole`**. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, le nom de rôle est sensible à la casse. Cependant, lorsqu'un nom de rôle apparaît aux clients dans la console, par exemple lors de la procédure d'ouverture de session, le nom de rôle est insensible à la casse.
> - Vous ne pouvez pas modifier le nom du rôle après sa création, car d'autres entités pourraient y faire référence.

10. (Facultatif) Pour Description, saisissez la description du rôle.
11. (Facultatif) Pour modifier les cas d'utilisation et les autorisations du rôle, dans les sections Étape 1 : sélectionner les entités de confiance ou Étape 2 : ajouter des autorisations, sélectionnez Modifier.
12. (Facultatif) Pour identifier, organiser ou rechercher le rôle, ajoutez des identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la section Balises pour les AWS Identity and Access Management ressources dans le Guide de l'utilisateur d'IAM.
13. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

Spécification de la AWS distribution pour le OpenTelemetry sidecar dans la définition de votre tâche

La console Amazon ECS simplifie l'expérience de création du conteneur AWS Distro for OpenTelemetry sidecar en utilisant l'option Use metric collection. Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.

Si vous n'utilisez pas la console Amazon ECS, vous pouvez ajouter manuellement le conteneur AWS Distro for OpenTelemetry sidecar à votre définition de tâche. L'exemple de définition de tâche suivant montre la définition de conteneur pour l'ajout de AWS Distro for OpenTelemetry sidecar pour l'intégration d'Amazon. CloudWatch

```
{
 "family": "otel-using-cloudwatch",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryCloudWatchRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [
    {
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
    }
   },
   "dependsOn": [{
    "containerName": "aws-otel-collector",
    "condition": "START"
   }]
  },
  {
   "name": "aws-otel-collector",
   "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
   "essential": true,
   "command": [
    "--config=/etc/ecs/ecs-cloudwatch.yaml"
   ],
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "True",
     "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
    }
   }
  }
 ],
 "networkMode": "awsvpc",
 "requiresCompatibilities": [
  "FARGATE"
```

```
 ],
 "cpu": "1024",
 "memory": "3072"
}
```

Exportation des métriques d'application vers Amazon Managed Service for Prometheus

Amazon ECS prend en charge l'exportation de vos métriques de processeur, de mémoire, de réseau et de stockage au niveau des tâches, ainsi que vos métriques d'application personnalisées vers Amazon Managed Service for Prometheus. Cela se fait en ajoutant le conteneur AWS Distro for OpenTelemetry sidecar à votre définition de tâche. La console Amazon ECS simplifie ce processus en ajoutant l'option Utiliser la collecte de métriques lors de la création d'une nouvelle définition de tâche. Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.

Les métriques sont exportées vers Amazon Managed Service for Prometheus et peuvent être consultées à l'aide du tableau de bord Amazon Managed Grafana. Votre application doit être instrumentée avec les bibliothèques Prometheus ou avec le SDK. OpenTelemetry Pour plus d'informations sur l'instrumentation de votre application avec le OpenTelemetry SDK, consultez la documentation Introduction à la AWS distribution pour OpenTelemetry in the AWS Distro. OpenTelemetry

Lorsque vous utilisez les bibliothèques Prometheus, votre application doit exposer un point de terminaison `/metrics` utilisé pour récupérer les données de métriques. Pour plus d'informations sur l'instrumentation de votre application avec les bibliothèques Prometheus, consultez Bibliothèques clientes Prometheus dans la documentation Prometheus.

Considérations

Les points suivants doivent être pris en compte lors de l'utilisation de l'intégration Amazon ECS on Fargate AWS avec Distro OpenTelemetry pour envoyer des métriques d'application à Amazon Managed Service for Prometheus.

- La AWS distribution pour l' OpenTelemetry intégration est prise en charge pour les charges de travail Amazon ECS hébergées sur Fargate et les charges de travail Amazon ECS hébergées sur des instances Amazon. EC2 Les instances externes ne sont actuellement pas prises en charge.
- Par défaut, AWS Distro for OpenTelemetry inclut toutes les dimensions de niveau tâche disponibles pour les métriques de votre application lors de l'exportation vers Amazon Managed

Service for Prometheus. Vous pouvez également instrumenter votre application pour ajouter des dimensions supplémentaires. Pour plus d'informations, consultez Getting Started with Prometheus Remote Write Exporter for Amazon Managed Service for Prometheus in the Distro pour obtenir de la documentation. AWS OpenTelemetry

Autorisations IAM requises pour AWS Distro pour OpenTelemetry l'intégration à Amazon Managed Service for Prometheus

L'intégration d'Amazon ECS à Amazon Managed Service for Prometheus à l'aide AWS de Distro OpenTelemetry pour sidecar nécessite que vous créiez un rôle IAM de tâche et que vous le spécifiiez dans la définition de votre tâche. Ce rôle IAM de tâche doit être créé manuellement en suivant les étapes ci-dessous avant d'enregistrer votre définition de tâche.

Nous recommandons que la AWS distribution pour OpenTelemetry sidecar soit également configurée pour acheminer les journaux des conteneurs vers les journaux, ce qui nécessite la création et la spécification d'un rôle IAM d'exécution de tâches dans votre définition de tâche. CloudWatch La console Amazon ECS prend en charge le rôle IAM d'exécution des tâches en votre nom, mais le rôle IAM des tâches doit être créé manuellement. Pour en savoir plus sur la création d'un rôle IAM d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS.

> **Important**
>
> Si vous collectez également des données de suivi d'applications à l'aide de AWS Distro à des fins OpenTelemetry d'intégration, assurez-vous que le rôle IAM de votre tâche contient également les autorisations nécessaires à cette intégration. Pour de plus amples informations, veuillez consulter Identifiez les opportunités d'optimisation d'Amazon ECS à l'aide des données de suivi des applications.

Pour créer le rôle de service pour Elastic Container Service (console IAM)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.

4. Pour Service ou cas d'utilisation, choisissez Elastic Container Service, puis choisissez le cas d'utilisation d'Elastic Container Service Task.
5. Choisissez Suivant.
6. Dans la section Ajouter des autorisations, recherchez AmazonPrometheusRemoteWriteAccess, puis sélectionnez la politique.
7. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée disponible pour les fonctions de service, mais pas pour les rôles liés à un service.

   a. Ouvrez la section Définir une limite des autorisations et choisissez Utiliser une limite des autorisations pour contrôler le nombre maximum d'autorisations de rôle.

      IAM inclut une liste des politiques AWS gérées et gérées par le client dans votre compte.

   b. Sélectionnez la politique à utiliser comme limite d'autorisations.
8. Choisissez Suivant.
9. Entrez un nom de rôle ou un suffixe de nom de rôle pour vous aider à identifier l'objectif du rôle.

> **Important**
>
> Lorsque vous nommez un rôle, notez ce qui suit :
>
> - Les noms de rôles doivent être uniques au sein du Compte AWS vôtre et ne peuvent pas être rendus uniques au cas par cas.
>
> Par exemple, ne créez pas deux rôles nommés **`PRODROLE`** et **`prodrole`**. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, le nom de rôle est sensible à la casse. Cependant, lorsqu'un nom de rôle apparaît aux clients dans la console, par exemple lors de la procédure d'ouverture de session, le nom de rôle est insensible à la casse.
> - Vous ne pouvez pas modifier le nom du rôle après sa création, car d'autres entités pourraient y faire référence.

10. (Facultatif) Pour Description, saisissez la description du rôle.
11. (Facultatif) Pour modifier les cas d'utilisation et les autorisations du rôle, dans les sections Étape 1 : sélectionner les entités de confiance ou Étape 2 : ajouter des autorisations, sélectionnez Modifier.

12. (Facultatif) Pour identifier, organiser ou rechercher le rôle, ajoutez des identifications sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la section Balises pour les AWS Identity and Access Management ressources dans le Guide de l'utilisateur d'IAM.
13. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

Spécification de la AWS distribution pour le OpenTelemetry sidecar dans la définition de votre tâche

La console Amazon ECS simplifie l'expérience de création du conteneur AWS Distro for OpenTelemetry sidecar en utilisant l'option Use metric collection. Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.

Si vous n'utilisez pas la console Amazon ECS, vous pouvez ajouter manuellement le conteneur AWS Distro for OpenTelemetry sidecar à votre définition de tâche. L'exemple de définition de tâche suivant montre la définition de conteneur pour l'ajout de l'intégration de AWS Distro for OpenTelemetry sidecar pour Amazon Managed Service for Prometheus.

```
{
 "family": "otel-using-cloudwatch",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryCloudWatchRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [{
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
     "awslogs-region": "aws-region",
     "awslogs-stream-prefix": "ecs"
    }
   },
   "dependsOn": [{
    "containerName": "aws-otel-collector",
    "condition": "START"
   }]
  },
  {
```

```
    "name": "aws-otel-collector",
    "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
    "essential": true,
    "command": [
     "--config=/etc/ecs/ecs-amp.yaml"
    ],
    "environment": [{
     "name": "AWS_PROMETHEUS_ENDPOINT",
     "value": "https://aps-workspaces.aws-region.amazonaws.com/workspaces/
ws-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/api/v1/remote_write"
    }],
    "logConfiguration": {
     "logDriver": "awslogs",
     "options": {
      "awslogs-create-group": "True",
      "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
      "awslogs-region": "aws-region",
      "awslogs-stream-prefix": "ecs"
     }
    }
   }
  ],
  "networkMode": "awsvpc",
  "requiresCompatibilities": [
   "FARGATE"
  ],
  "cpu": "1024",
  "memory": "3072"
}
```

Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail

Amazon Elastic Container Service est intégré à AWS CloudTrailun service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un Service AWS. CloudTrail capture tous les appels d'API pour Amazon ECS sous forme d'événements. Les appels capturés incluent des appels provenant de la console Amazon ECS et des appels de code vers les opérations de l'API Amazon ECS. À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande envoyée à Amazon ECS, l'adresse IP à partir de laquelle la demande a été faite, la date à laquelle elle a été faite et des informations supplémentaires.

Chaque événement ou entrée de journal contient des informations sur la personne ayant initié la demande. Les informations relatives à l'identité permettent de déterminer :

- Si la demande a été effectuée avec des informations d'identification d'utilisateur root ou d'utilisateur root.
- Si la demande a été faite au nom d'un utilisateur du centre d'identité IAM.
- Si la demande a été effectuée avec les informations d'identification de sécurité temporaires d'un rôle ou d'un utilisateur fédéré.
- Si la requête a été effectuée par un autre Service AWS.

CloudTrail est actif dans votre compte Compte AWS lorsque vous créez le compte et vous avez automatiquement accès à l'historique des CloudTrail événements. L'historique des CloudTrail événements fournit un enregistrement consultable, consultable, téléchargeable et immuable des 90 derniers jours des événements de gestion enregistrés dans un. Région AWS Pour plus d'informations, consultez la section Utilisation de l'historique des CloudTrail événements dans le guide de AWS CloudTrail l'utilisateur. La consultation de CloudTrail l'historique des événements est gratuite.

Pour un enregistrement continu des événements de vos 90 Compte AWS derniers jours, créez un magasin de données sur les événements de Trail ou CloudTrailLake.

CloudTrail sentiers

Un suivi permet CloudTrail de fournir des fichiers journaux à un compartiment Amazon S3. Tous les sentiers créés à l'aide du AWS Management Console sont multirégionaux. Vous ne pouvez créer un journal de suivi en une ou plusieurs régions à l'aide de l' AWS CLI. Il est recommandé de créer un parcours multirégional, car vous capturez l'activité dans l'ensemble Régions AWS de votre compte. Si vous créez un journal de suivi pour une seule région, il convient de n'afficher que les événements enregistrés dans le journal de suivi pour une seule région Région AWS. Pour plus d'informations sur les journaux de suivi, consultez Créez un journal de suivi dans vos Compte AWS et Création d'un journal de suivi pour une organisation dans le AWS CloudTrail Guide de l'utilisateur.

Vous pouvez envoyer une copie de vos événements de gestion en cours dans votre compartiment Amazon S3 gratuitement CloudTrail en créant un journal. Toutefois, des frais de stockage Amazon S3 sont facturés. Pour plus d'informations sur la CloudTrail tarification, consultez la section AWS CloudTrail Tarification. Pour obtenir des informations sur la tarification Amazon S3, consultez Tarification Amazon S3.

CloudTrail Stockages de données sur les événements du lac

CloudTrail Lake vous permet d'exécuter des requêtes SQL sur vos événements. CloudTrail Lake convertit les événements existants au format JSON basé sur les lignes au format Apache ORC. ORC est un format de stockage en colonnes qui est optimisé pour une récupération rapide des données. Les événements sont agrégés dans des magasins de données d'événement. Ceux-ci constituent des collections immuables d'événements basées sur des critères que vous sélectionnez en appliquant des sélecteurs d'événements avancés. Les sélecteurs que vous appliquez à un magasin de données d'événement contrôlent les événements qui persistent et que vous pouvez interroger. Pour plus d'informations sur CloudTrail Lake, consultez la section Travailler avec AWS CloudTrail Lake dans le guide de AWS CloudTrail l'utilisateur.

CloudTrail Les stockages et requêtes de données sur les événements de Lake entraînent des coûts. Lorsque vous créez un magasin de données d'événement, vous choisissez l'option de tarification que vous voulez utiliser pour le magasin de données d'événement. L'option de tarification détermine le coût d'ingestion et de stockage des événements, ainsi que les périodes de conservation par défaut et maximale pour le magasin de données d'événement. Pour plus d'informations sur la CloudTrail tarification, consultez la section AWS CloudTrail Tarification.

Événements de gestion Amazon ECS dans CloudTrail

Les événements de gestion fournissent des informations sur les opérations de gestion effectuées sur les ressources de votre Compte AWS. Ils sont également connus sous le nom opérations de plan de contrôle. Par défaut, CloudTrail enregistre les événements de gestion.

Amazon Elastic Container Service enregistre toutes les opérations du plan de contrôle Amazon ECS en tant qu'événements de gestion. Par exemple, les appels aux `DeleteCluster` sections`CreateService`, `RunTask` et génèrent des entrées dans les fichiers CloudTrail journaux. Pour obtenir la liste des opérations du plan de contrôle Amazon Elastic Container Service auxquelles Amazon ECS se connecte CloudTrail, consultez le manuel Amazon Elastic Container Service API Reference.

Exemples d'événements Amazon ECS

Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'opération d'API demandée, la date et l'heure de l'opération, les paramètres de la demande, etc. CloudTrail les fichiers journaux ne constituent pas une trace ordonnée des appels d'API publics. Les événements n'apparaissent donc pas dans un ordre spécifique.

L'exemple suivant montre un CloudTrail événement illustrant l'`CreateService`action.

> **Note**
>
> Cet exemple a été mis en forme pour faciliter sa lecture. Dans un fichier CloudTrail journal, toutes les entrées et tous les événements sont concaténés sur une seule ligne. En outre, cet exemple se limite à une seule entrée Amazon ECS. Dans un véritable fichier CloudTrail journal, vous pouvez voir les entrées et les événements de plusieurs AWS services.

```
{
    "eventVersion": "1.04",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:account_name",
        "arn": "arn:aws:sts::123456789012:user/Mary_Major",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-06-20T18:32:25Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Mary_Major"
            }
        }
    },
    "eventTime": "2018-06-20T19:04:36Z",
    "eventSource": "ecs.amazonaws.com",
    "eventName": "CreateCluster",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.12",
    "userAgent": "console.amazonaws.com",
    "requestParameters": {
        "clusterName": "default"
    },
    "responseElements": {
```

```
        "cluster": {
            "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/default",
            "pendingTasksCount": 0,
            "registeredContainerInstancesCount": 0,
            "status": "ACTIVE",
            "runningTasksCount": 0,
            "statistics": [],
            "clusterName": "default",
            "activeServicesCount": 0
        }
    },
    "requestID": "cb8c167e-EXAMPLE",
    "eventID": "e3c6f4ce-EXAMPLE",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

Pour plus d'informations sur le contenu des CloudTrail enregistrements, voir le contenu des CloudTrail enregistrements dans le Guide de AWS CloudTrail l'utilisateur.

Affichage CloudWatch des journaux en direct pour les services et tâches Amazon ECS

CloudWatch Logs Live Tail vous aide à résoudre rapidement les incidents en visualisant la liste des nouveaux événements enregistrés au fur et à mesure de leur ingestion. Vous pouvez consulter les événements relatifs aux tâches et aux services dans la console Amazon ECS. Cette vue fournit une vue cohérente de l'état de vos tâches et de vos services.

Chaque tâche d'un service Amazon ECS possède des flux de journaux dédiés pour chaque conteneur. Si un service prend de l'ampleur, chaque instance de tâche possède son propre ensemble de flux de journaux. La convention de dénomination des flux de journaux suit le modèle suivant :

```
log-group-prefix/container-name/task-id
```

Une seule tâche écrit dans les mêmes flux de log au cours de sa durée de vie. Le flux de log contient les messages provenant des conteneurs de cette tâche ainsi que toute sortie du code de votre application. Chaque message est horodaté, y compris vos journaux personnalisés.

> **Note**
>
> Le coût des sessions Live Tail est calculé en fonction du temps d'utilisation de la session, par minute. Pour plus d'informations sur les tarifs, consultez Amazon CloudWatch Pricing.

Autorisations requises

Les autorisations suivantes sont requises pour que l'utilisateur IAM de la console puisse démarrer et arrêter les sessions CloudWatch Logs Live Tail :

- `logs:DescribeLogGroups`
- `logs:StartLiveTail`
- `logs:StopLiveTail`

Procédure

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Déterminez la ressource que vous souhaitez consulter

Ressource	Étapes
Tâches	a. Sur la page Clusters, choisissez le cluster. La page de détails du cluster s'affiche. b. Choisissez l'onglet Tasks. La page des détails de la tâche s'affiche. c. Sélectionnez l'onglet Logs (Journaux).
Services	a. Sur la page Clusters, choisissez le cluster.

Ressource	Étapes	
	La page de détails du cluster s'affiche. b. Choisissez le service. La page des détails du service s'affiche. c. Sélectionnez l'onglet Logs (Journaux).	

3. Choisissez CloudWatch Logs Live Tail, puis Start.
4. (Facultatif) Pour filtrer les flux, sous Filtrer, pour Sélectionner les groupes de journaux, choisissez le groupe de journaux.

AWS CLI références

Vous pouvez également utiliser le AWS CLI pour démarrer des sessions Live Tail pour CloudWatch Logs. Les ressources suivantes fournissent des informations détaillées sur l'utilisation AWS CLI de Live Tail :

- start-live-tail référence de commande - Syntaxe complète de commande, paramètres et exemples de `aws logs start-live-tail` commande.
- CloudWatch Guide de l'utilisateur de Logs Live Tail - Guide complet incluant AWS CLI l'utilisation avec les modes d'impression uniquement et interactif.
- StartLiveTail Exemples de SDK - Exemples programmatiques d'utilisation de l' StartLiveTail API avec différents. AWS SDKs

Surveillez les charges de travail à l'aide des métadonnées Amazon ECS

Vous pouvez utiliser les métadonnées des tâches et des conteneurs pour résoudre les problèmes liés à vos charges de travail et apporter des modifications de configuration en fonction de l'environnement d'exécution.

Les métadonnées incluent les catégories suivantes :

- Attributs au niveau de la tâche qui fournissent des informations sur le lieu d'exécution de la tâche.
- Attributs au niveau du conteneur qui fournissent l'ID, le nom et les détails de l'image Docker.

  Cela donne de la visibilité à l'intérieur du conteneur.
- Paramètres réseau tels que les adresses IP, les sous-réseaux et le mode réseau.

  Cela facilite la configuration et le dépannage du réseau.
- État et santé des tâches

  Cela vous permet de savoir si les tâches sont en cours d'exécution.

Vous pouvez afficher les métadonnées par l'une des méthodes suivantes :

- Fichier de métadonnées de conteneur

  À compter de la version 1.15.0 de l'agent conteneur Amazon ECS, diverses métadonnées de conteneur sont disponibles dans les conteneurs ou dans l'instance de conteneur hôte. En activant cette fonctionnalité, vous pouvez demander des informations sur une tâche, un conteneur ou une instance de conteneur depuis le conteneur ou l'instance de conteneur hôte. Le fichier de métadonnées est créé sur l'instance hôte et monté dans le conteneur en tant que volume Docker et n'est donc pas disponible lorsqu'une tâche est hébergée sur AWS Fargate.
- Point de terminaison des métadonnées de tâches

  L'agent de conteneur Amazon ECS injecte une variable d'environnement dans chaque conteneur, appelée point de terminaison des métadonnées de tâches, qui fournit diverses métadonnées de tâches et statistiques Docker au conteneur.
- Introspection des conteneurs

  L'agent de conteneur Amazon ECS fournit une opération API pour l'obtention d'informations sur l'instance de conteneur sur laquelle l'agent est exécuté et sur les tâches connexes elles aussi exécutées sur cette instance.

Variables d'environnement Amazon ECS

Les variables d'environnement contrôlent le comportement d'une tâche, notamment la capacité utilisée pour exécuter la tâche. Amazon ECS définit les variables d'environnement suivantes pour vos tâches :

- `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI`- Le point de terminaison URL HTTP complet que le SDK doit utiliser lors d'une demande d'informations d'identification. Cela inclut à la fois le schéma et l'hôte. Pour plus d'informations, consultez la section Fournisseur d'informations d'identification du conteneur dans le Guide de AWS SDKs référence.
- `ECS_CONTAINER_METADATA_URI_V4`- L'adresse de la version 4 des métadonnées de la tâche. Pour de plus amples informations, veuillez consulter Point de terminaison des métadonnées des tâches Amazon ECS, version 4.
- `ECS_CONTAINER_METADATA_URI`- L'adresse de la version 3 des métadonnées de la tâche. Pour de plus amples informations, veuillez consulter Point de terminaison des métadonnées des tâches Amazon ECS, version 3.
- `ECS_AGENT_URI`- L'adresse de base pour les différents points de terminaison pris en charge par Fargate. Pour plus d'informations, consultez :
  - Point de terminaison de protection évolutif des tâches Amazon ECS
  - Points de terminaison d'injection de pannes Amazon ECS
- `AWS_EXECUTION_ENV`- Les informations relatives au type de lancement sur lequel la tâche est exécutée.
  - Pour Fargate, Amazon ECS définit ce paramètre sur. `AWS_ECS_FARGATE`
  - Pour EC2, Amazon ECS définit ce paramètre sur`AWS_ECS_EC2`.
- `AWS_DEFAULT_REGION`— La valeur par défaut Région AWS pour votre Compte AWS. Il s'agit de la région par défaut dans laquelle votre tâche est exécutée.
- `AWS_REGION`— La région dans laquelle la tâche est exécutée. Si elle est définie, cette valeur remplace `AWS_DEFAULT_REGION`.

Vous pouvez consulter les variables d'environnement dans les métadonnées des tâches. Pour plus d'informations, consultez l'une des rubriques suivantes :

- Fargate - Métadonnées des tâches Amazon ECS disponibles pour les tâches sur Fargate
- EC2 - Métadonnées de tâches disponibles pour les tâches Amazon ECS sur EC2

Fichier de métadonnées de conteneur Amazon ECS

À compter de la version 1.15.0 de l'agent conteneur Amazon ECS, diverses métadonnées de conteneur sont disponibles dans les conteneurs ou dans l'instance de conteneur hôte. En activant cette fonctionnalité, vous pouvez demander des informations sur une tâche, un conteneur ou

une instance de conteneur depuis le conteneur ou l'instance de conteneur hôte. Le fichier de métadonnées est créé sur l'instance hôte et monté dans le conteneur en tant que volume Docker. Il n'est donc pas disponible lorsqu'une tâche est hébergée sur AWS Fargate.

Le fichier de métadonnées de conteneur est nettoyé sur l'instance de l'hôte lors du nettoyage du conteneur. Vous pouvez définir quand cette opération doit se produire avec la variable `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` de l'agent de conteneur. Pour de plus amples informations, veuillez consulter Nettoyage automatique des tâches et des images Amazon ECS.

Rubriques

- Emplacements des fichiers de métadonnées de conteneur
- Activation des métadonnées du conteneur Amazon ECS
- Format de fichier de métadonnées de conteneur Amazon ECS

Emplacements des fichiers de métadonnées de conteneur

Par défaut, le fichier de métadonnées de conteneur est écrit sur les chemins d'hôte et de conteneur suivants.

- Pour les instances Linux :
  - Chemin hôte :`/var/lib/ecs/data/metadata/`*`cluster_name`*`/`*`task_id`*`/`*`container_name`*`/ecs-container-metadata.json`

    > **Note**
    >
    > Le chemin hôte Linux suppose que le chemin de montage du répertoire de données par défaut (`/var/lib/ecs/data`) est utilisé lorsque l'agent est démarré. Si vous n'utilisez pas l'AMI optimisée pour Amazon ECS (ou le package `ecs-init` pour démarrer et gérer l'agent de conteneur), veillez à définir la variable de configuration de l'agent `ECS_HOST_DATA_DIR` sur le chemin hôte où se trouve le fichier d'état de l'agent de conteneur. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

  - Chemin du conteneur : `/opt/ecs/metadata/`*`random_ID`*`/ecs-container-metadata.json`
- Pour les instances Windows :

- Chemin hôte :`C:\ProgramData\Amazon\ECS\data\metadata\task_id\container_name\ecs-container-metadata.json`
- Chemin du conteneur : `C:\ProgramData\Amazon\ECS\metadata\random_ID\ecs-container-metadata.json`

Toutefois, pour en faciliter l'accès, l'emplacement du fichier de métadonnées de conteneur correspond à la variable d'environnement `ECS_CONTAINER_METADATA_FILE` à l'intérieur du conteneur. Vous pouvez lire le contenu du fichier depuis le conteneur via la commande suivante :

- Pour les instances Linux :

```
cat $ECS_CONTAINER_METADATA_FILE
```

- Pour les instances Windows (PowerShell) :

```
Get-Content -path $env:ECS_CONTAINER_METADATA_FILE
```

Activation des métadonnées du conteneur Amazon ECS

Vous pouvez activer les métadonnées de conteneur au niveau de l'instance de conteneur en définissant la variable `ECS_ENABLE_CONTAINER_METADATA` de l'agent de conteneur sur `true`. Vous pouvez spécifier cette variable dans le fichier de configuration `/etc/ecs/ecs.config` et redémarrer l'agent. Vous pouvez également la configurer sous forme de variable d'environnement Docker au moment de l'exécution lorsque l'agent de conteneur est démarré. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

Si la valeur `ECS_ENABLE_CONTAINER_METADATA` est définie sur `true` au démarrage de l'agent, des fichiers de métadonnées sont créés pour tous les conteneurs générés à partir de cet instant. L'agent de conteneur Amazon ECS ne peut pas créer de fichiers de métadonnées pour les conteneurs qui ont été créés avant que la variable d'agent de conteneur `ECS_ENABLE_CONTAINER_METADATA` n'ait été définie sur `true`. Pour vous assurer que tous les conteneurs reçoivent des fichiers de métadonnées, vous devez définir cette variable d'agent au lancement de l'instance de conteneur. Voici un exemple de script de données utilisateur qui définit cette variable et enregistre votre instance de conteneur auprès de votre cluster.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
```

```
ECS_CLUSTER=your_cluster_name
ECS_ENABLE_CONTAINER_METADATA=true
EOF
```

Format de fichier de métadonnées de conteneur Amazon ECS

Les informations suivantes sont stockées dans le fichier JSON de métadonnées de conteneur.

`Cluster`

Nom du cluster sur lequel le conteneur de la tâche est en cours d'exécution.

`ContainerInstanceARN`

Amazon Resource Name (ARN) de l'instance de conteneur hôte.

`TaskARN`

Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`TaskDefinitionFamily`

Nom de la famille de définitions de tâche utilisée par le conteneur.

`TaskDefinitionRevision`

Révision de définition de tâche utilisée par le conteneur.

`ContainerID`

ID de conteneur Docker (et non l'ID de conteneur Amazon ECS).

`ContainerName`

Nom de conteneur issu de la définition de tâche Amazon ECS.

`DockerContainerName`

Nom de conteneur que le démon Docker utilise pour le conteneur (par exemple, nom qui s'affiche dans la sortie de la commande docker ps).

`ImageID`

Résumé SHA de l'image Docker utilisée pour lancer le conteneur.

`ImageName`

Nom et balise de l'image Docker utilisée pour lancer le conteneur.

`PortMappings`

Mappages de port associés au conteneur.

`ContainerPort`

Port du conteneur qui est exposé.

`HostPort`

Port de l'instance de conteneur hôte qui est exposé.

`BindIp`

Adresse IP de liaison affectée au conteneur par Docker. Cette adresse IP est uniquement appliquée avec le mode réseau `bridge`. Elle est uniquement accessible à partir de l'instance de conteneur.

`Protocol`

Protocole réseau utilisé pour le mappage de port.

`Networks`

Mode réseau et adresse IP du conteneur.

`NetworkMode`

Mode réseau de la tâche à laquelle le conteneur appartient.

`IPv4Addresses`

Adresses IP associées au conteneur.

> **Important**
>
> Si votre tâche utilise le mode réseau `awsvpc`, l'adresse IP du conteneur n'est pas renvoyée. Dans ce cas, vous pouvez récupérer l'adresse IP en lisant le fichier /etc/hosts à l'aide de la commande suivante :
>
> ```
> tail -1 /etc/hosts | awk '{print $1}'
> ```

`MetadataFileStatus`

Statut du fichier de métadonnées. Lorsque le statut est `READY`, le fichier de métadonnées est actuel et complet. Si le fichier n'est pas encore prêt (par exemple, au moment où la tâche est démarrée), une version tronquée du format de fichier est disponible. Pour éviter toute condition de concurrence où le conteneur démarrerait alors que les métadonnées ne sont pas encore écrites, vous pouvez analyser le fichier de métadonnées et attendre que ce paramètre passe à `READY` au préalable, selon les métadonnées. Cela est généralement disponible en moins de 1 seconde à partir du moment où le conteneur démarre.

`AvailabilityZone`

Zone de disponibilité dans laquelle réside l'instance de conteneur hôte.

`HostPrivateIPv4Address`

Adresse IP privée de la tâche à laquelle appartient le conteneur.

`HostPublicIPv4Address`

Adresse IP publique de la tâche à laquelle appartient le conteneur.

Example Fichier de métadonnées de conteneur Amazon ECS (**`READY`**)

L'exemple suivant montre un conteneur de fichiers de métadonnées avec le statut `READY`.

```
{
    "Cluster":"arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
    "TaskARN":"arn:aws:ecs:us-east-1:123456789012:task/MyCluster/
b593651c4d6b44a6b2b583f45c957e15",
    "Family":"curltest-container",
    "Revision":"2",
    "DesiredStatus":"RUNNING",
    "KnownStatus":"RUNNING",
    "Limits":
        {
            "CPU":0.25,
            "Memory":512
        },
    "PullStartedAt":"2025-01-17T20:56:17.394610044Z",
    "PullStoppedAt":"2025-01-17T20:56:25.282708213Z",
    "AvailabilityZone":"us-east-1b",
    "LaunchType":"FARGATE",
    "Containers":[
```

```
        {
            "DockerId":"b593651c4d6b44a6b2b583f45c957e15-3356213583",
            "Name":"curltest","DockerName":"curltest",
            "Image":"public.ecr.aws/amazonlinux/amazonlinux:latest",

 "ImageID":"sha256:7f371357694782356b65c7fd60dd1ca124c47bd5ed1b1ffe7c0e17f562898367",
            "Labels":
                {
                    "com.amazonaws.ecs.cluster":"arn:aws:ecs:us-
east-1:123456789012:cluster/MyCluster",
                    "com.amazonaws.ecs.container-name":"curltest",
                    "com.amazonaws.ecs.task-arn":"arn:aws:ecs:us-
east-1:123456789012:task/MyCluster/b593651c4d6b44a6b2b583f45c957e15",
                    "com.amazonaws.ecs.task-definition-family":"curltest-
container","com.amazonaws.ecs.task-definition-version":"2"
                },
            "DesiredStatus":"RUNNING",
            "KnownStatus":"RUNNING",
            "Limits":
                {
                    "CPU":2
                },
            "CreatedAt":"2025-01-17T20:56:26.180347056Z",
            "StartedAt":"2025-01-17T20:56:26.180347056Z",
            "Type":"NORMAL",
            "LogDriver":"awslogs",
            "LogOptions":
                {
                    "awslogs-create-group":"true",
                    "awslogs-group":"/ecs/curltest-container",
                    "awslogs-region":"us-east-1",
                    "awslogs-stream":"ecs/curltest/b593651c4d6b44a6b2b583f45c957e15"
                       },
            "ContainerARN":"arn:aws:ecs:us-east-1:123456789012:container/MyCluster/
b593651c4d6b44a6b2b583f45c957e15/934575e8-5bdb-478f-b763-2341a85b690e",
            "Networks":[
                {
                    "NetworkMode":"awsvpc",
                    "IPv4Addresses":["10.0.1.58"]
                }
            ],
            "Snapshotter":"overlayfs"
        }
    ],
```

```
    "ClockDrift":
        {
            "ClockErrorBound":0.487801,"ReferenceTimestamp":"2025-01-17T20:56:02Z",
            "ClockSynchronizationStatus":"SYNCHRONIZED"
        },
    "FaultInjectionEnabled":false
}
```

Example Fichier de métadonnées de conteneur Amazon ECS incomplet (pas encore **READY**)

L'exemple suivant montre un fichier de métadonnées de conteneur qui n'a pas encore atteint le statut READY. Les informations contenues dans le fichier sont limitées à quelques paramètres connus à partir de la définition de tâche. Le fichier de métadonnées de conteneur doit être prêt 1 seconde à compter du démarrage du conteneur.

```
{
    "Cluster": "default",
    "ContainerInstanceARN": "arn:aws:ecs:us-west-2:012345678910:container-instance/
default/1f73d099-b914-411c-a9ff-81633b7741dd",
    "TaskARN": "arn:aws:ecs:us-west-2:012345678910:task/default/
d90675f8-1a98-444b-805b-3d9cabb6fcd4",
    "ContainerName": "metadata"
}
```

Métadonnées de tâches disponibles pour les tâches Amazon ECS sur EC2

L'agent de conteneur Amazon ECS fournit une méthode pour récupérer les diverses métadonnées de tâches et les statistiques Docker. On parle alors du point de terminaison des métadonnées de tâches. Les versions suivantes sont disponibles :

- Point de terminaison des métadonnées de tâche version 4 : fournit une variété de métadonnées et de statistiques Docker aux conteneurs. Peut également fournir des données de débit réseau. Disponible pour les tâches Amazon ECS lancées sur des instances Amazon EC2 Linux exécutant au moins une version `1.39.0` de l'agent de conteneur Amazon ECS. Pour les instances Amazon EC2 Windows qui utilisent le mode `awsvpc` réseau, l'agent de conteneur Amazon ECS doit être au moins une version`1.54.0`. Pour de plus amples informations, veuillez consulter Point de terminaison des métadonnées des tâches Amazon ECS, version 4.
- Point de terminaison des métadonnées de tâche version 3 : fournit une variété de métadonnées et de statistiques Docker aux conteneurs. Disponible pour les tâches Amazon ECS lancées sur des instances Amazon EC2 Linux exécutant au moins une version `1.21.0` de l'agent de conteneur

Amazon ECS. Pour les instances Amazon EC2 Windows qui utilisent le mode `awsvpc` réseau, l'agent de conteneur Amazon ECS doit être au moins une version`1.54.0`. Pour de plus amples informations, veuillez consulter Point de terminaison des métadonnées des tâches Amazon ECS, version 3.

- Version 2 du point de terminaison des métadonnées des tâches : disponible pour les tâches Amazon ECS lancées sur des instances Amazon EC2 Linux exécutant au moins une version `1.17.0` de l'agent de conteneur Amazon ECS. Pour les instances Amazon EC2 Windows qui utilisent le mode `awsvpc` réseau, l'agent de conteneur Amazon ECS doit être au moins une version`1.54.0`. Pour de plus amples informations, veuillez consulter Point de terminaison des métadonnées des tâches Amazon ECS, version 2.

Si votre tâche Amazon ECS est hébergée sur Amazon EC2, vous pouvez également accéder aux métadonnées de l'hôte des tâches à l'aide du point de terminaison IMDS (Instance Metadata Service). La commande suivante, lorsqu'elle est exécutée depuis l'instance hébergeant la tâche, répertorie l'ID de l'instance hôte.

```
curl http://169.254.169.254/latest/meta-data/instance-id
```

Les informations que vous pouvez obtenir à partir du point de terminaison sont divisées en catégories telles que *`instance-id`*. Pour plus d'informations sur les différentes catégories de métadonnées d'instance hôte que vous pouvez obtenir à l'aide du point de terminaison, consultez la section Catégories de métadonnées d'instance.

Point de terminaison des métadonnées des tâches Amazon ECS, version 4

L'agent de conteneur Amazon ECS injecte une variable d'environnement dans chaque conteneur, appelée point de terminaison des métadonnées de tâches, qui fournit diverses métadonnées de tâches et statistiques Docker au conteneur.

Les métadonnées des tâches et les statistiques du débit du réseau sont envoyées à CloudWatch Container Insights et peuvent être consultées dans le AWS Management Console. Pour de plus amples informations, veuillez consulter Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée.

> **Note**
>
> Amazon ECS fournit des versions antérieures du point de terminaison des métadonnées de tâches. Pour éviter de créer d'autres versions de point de terminaison de métadonnées

de tâche à l'avenir, des métadonnées supplémentaires peuvent être ajoutées à la sortie de la version 4. Nous ne supprimons pas les métadonnées existantes et ne modifions pas les noms des champs de métadonnées.

La variable d'environnement est injectée par défaut dans les conteneurs des tâches Amazon ECS lancées sur des instances Amazon EC2 Linux qui exécutent au moins une version `1.39.0` de l'agent de conteneur Amazon ECS. Pour les instances Amazon EC2 Windows qui utilisent le mode `awsvpc` réseau, l'agent de conteneur Amazon ECS doit être au moins une version`1.54.0`. Pour de plus amples informations, veuillez consulter Gestion des instances de conteneurs Linux Amazon ECS.

> **Note**
>
> Vous pouvez ajouter la prise en charge de cette fonctionnalité sur les EC2 instances Amazon utilisant d'anciennes versions de l'agent de conteneur Amazon ECS en mettant à jour l'agent vers la dernière version. Pour de plus amples informations, veuillez consulter Mise à jour de l'agent de conteneur Amazon ECS.

Pour des exemples de sortie de métadonnées de tâches, consultez les exemples de métadonnées de tâches Amazon ECS v4.

Chemins des points de terminaison des métadonnées de tâches version 4

Les chemins des points de terminaison des métadonnées de tâches suivants sont disponibles pour les conteneurs :

`${ECS_CONTAINER_METADATA_URI_V4}`

Ce chemin renvoie des métadonnées pour le conteneur.

`${ECS_CONTAINER_METADATA_URI_V4}/task`

Ce chemin renvoie les métadonnées de la tâche, y compris une liste des conteneurs IDs et les noms de tous les conteneurs associés à la tâche. Pour plus d'informations sur la réponse pour ce point de terminaison, consultez Réponse JSON V4 aux métadonnées des tâches Amazon ECS.

`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`

Ce chemin renvoie les métadonnées de la tâche incluse dans le point de terminaison `/task` en plus des balises de tâche et d'instance de conteneur qui peuvent être récupérées à l'aide de l'API

`ListTagsForResource`. Toutes les erreurs reçues lors de la récupération des métadonnées de balise seront incluses dans le champ `Errors` de la réponse.

> **Note**
>
> Le `Errors` champ figure uniquement dans la réponse pour les tâches hébergées sur des instances Amazon EC2 Linux exécutant au moins une version `1.50.0` de l'agent de conteneur. Pour les instances Amazon EC2 Windows qui utilisent le mode `awsvpc` réseau, l'agent de conteneur Amazon ECS doit être au moins la version `1.54.0`
> Ce point de terminaison nécessite l'autorisation `ecs.ListTagsForResource`.

> **Important**
>
> Lorsque vous utilisez le `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags` point de terminaison, sachez que chaque appel génère jusqu'à deux demandes d'API `ecs:ListTagsForResource` (une pour les balises d'instance de conteneur et une pour les balises de tâche) et que tous les conteneurs annexes de la tâche peuvent effectuer ces appels en votre nom. Les appels fréquents aux terminaux peuvent entraîner une limitation de l'API.
> Envisagez de mettre en œuvre des stratégies de mise en cache ou de traitement par lots pour réduire la fréquence des appels, en particulier dans les applications à fort trafic, et de résoudre les problèmes de limitation des API en utilisant. AWS CloudTrail Pour plus d'informations sur les limites de limitation de l'`ListTagsForResource`API, consultez la section Régulation des demandes pour l'API Amazon ECS dans le manuel Amazon Elastic Container Service API Reference. Pour plus d'informations sur le débogage des appels d'API Amazon ECS à l'aide AWS CloudTrail de. Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail

`${ECS_CONTAINER_METADATA_URI_V4}/stats`

Ce chemin renvoie les statistiques Docker pour ce conteneur spécifique. Pour plus d'informations sur chacune des statistiques renvoyées, consultez la documentation ContainerStatsde l'API Docker.

Pour les tâches Amazon ECS qui utilisent le mode `bridge` réseau `awsvpc` ou le mode réseau hébergés sur des instances Amazon EC2 Linux exécutant au moins une version `1.43.0` de

l'agent de conteneur, des statistiques supplémentaires sur le débit du réseau seront incluses dans la réponse. Pour toutes les autres tâches, la réponse inclura uniquement les statistiques cumulatives du réseau.

`${ECS_CONTAINER_METADATA_URI_V4}/task/stats`

Ce chemin renvoie les statistiques Docker pour tous les conteneurs associés à la tâche. Il peut être utilisé par les conteneurs sidecar pour extraire les métriques réseau. Pour plus d'informations sur chacune des statistiques renvoyées, consultez la documentation ContainerStatsde l'API Docker.

Pour les tâches Amazon ECS qui utilisent le mode `bridge` réseau `awsvpc` ou le mode réseau hébergés sur des instances Amazon EC2 Linux exécutant au moins une version `1.43.0` de l'agent de conteneur, des statistiques supplémentaires sur le débit du réseau seront incluses dans la réponse. Pour toutes les autres tâches, la réponse inclura uniquement les statistiques cumulatives du réseau.

Réponse JSON V4 aux métadonnées des tâches Amazon ECS

Les informations suivantes sont renvoyées à partir de la réponse JSON du point de terminaison de métadonnées de la tâche (`${ECS_CONTAINER_METADATA_URI_V4}/task`). Cela inclut les métadonnées associées à la tâche en plus des métadonnées de chaque conteneur dans la tâche.

`Cluster`

Amazon Resource Name (ARN) ou nom abrégé du cluster Amazon ECS auquel la tâche appartient.

`ServiceName`

Nom du service auquel appartient la tâche. ServiceName apparaîtra pour les instances de conteneur Amazon EC2 et Amazon ECS Anywhere si la tâche est associée à un service.

> **Note**
>
> Les métadonnées `ServiceName` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS `1.63.1` ou version ultérieure.

`VPCID`

L'ID VPC de l'instance de EC2 conteneur Amazon. Ce champ n'apparaît que pour les EC2 instances Amazon.

> **Note**
>
> Les métadonnées `VPCID` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS `1.63.1` ou version ultérieure.

`TaskARN`

Le nom de ressource Amazon (ARN) de la tâche à laquelle appartient le conteneur.

`Family`

Famille de la définition de tâche Amazon ECS pour la tâche.

`Revision`

Révision de la définition de tâche Amazon ECS pour la tâche.

`DesiredStatus`

État souhaité pour la tâche à partir d'Amazon ECS.

`KnownStatus`

État connu pour la tâche à partir d'Amazon ECS.

`Limits`

Les limites de ressources spécifiées au niveau de la tâche, telles que le processeur (exprimé en vCPUs) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`PullStartedAt`

Horodatage au moment où la première extraction de l'image de conteneur a commencé.

`PullStoppedAt`

Horodatage au moment où la dernière extraction de l'image de conteneur s'est terminée.

`AvailabilityZone`

Zone de disponibilité dans laquelle se trouve la tâche.

> **Note**
>
> Les métadonnées de la zone de disponibilité sont uniquement disponibles pour les tâches Fargate qui utilisent la version 1.4 de la plateforme ou ultérieure (Linux) ou la version 1.0.0 de la plateforme (Windows).

`LaunchType`

Type de lancement utilisé par la tâche. Lorsque vous utilisez des fournisseurs de capacité de cluster, cela indique si la tâche utilise EC2 Fargate ou une infrastructure.

> **Note**
>
> Ces métadonnées `LaunchType` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure (Linux) ou 1.0.0 ou ultérieure (Windows).

`Containers`

Liste de métadonnées de conteneur pour chaque conteneur associé à la tâche.

`DockerId`

ID Docker du conteneur.

Lorsque vous utilisez Fargate, l'ID est un hexadécimal de 32 chiffres suivi d'un nombre de 10 chiffres.

`Name`

Nom du conteneur tel que spécifié dans la définition de tâche.

`DockerName`

Nom du conteneur fourni à Docker. L'agent de conteneur Amazon ECS génère un nom unique pour le conteneur afin d'éviter les conflits de noms lorsque plusieurs copies de la même définition de tâche sont exécutées sur une seule instance.

`Image`

Image à utiliser pour le conteneur.

`ImageID`

Le condensé SHA-256 du manifeste d'image. Il s'agit du condensé qui peut être utilisé pour extraire l'image en utilisant le format`repository-url/image@sha256:digest`.

`Ports`

Tous les ports exposés pour le conteneur. Ce paramètre n'est pas spécifié s'il n'y a pas de ports exposés.

`Labels`

Toutes les étiquettes appliquées au conteneur. Ce paramètre n'est pas spécifié si aucune étiquette n'est appliquée.

`DesiredStatus`

État souhaité pour le conteneur à partir d'Amazon ECS.

`KnownStatus`

État connu pour le conteneur à partir d'Amazon ECS.

`ExitCode`

Code de sortie pour le conteneur. Ce paramètre n'est pas spécifié si le conteneur n'est pas sorti.

`Limits`

Limites de ressource spécifiées au niveau du conteneur, par exemple, le processeur (exprimé en unités de processeur) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`CreatedAt`

Horodatage de création du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été créé.

`StartedAt`

Horodatage de démarrage du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été démarré.

`FinishedAt`

Horodatage d'arrêt du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été arrêté.

`Type`

Type du conteneur. Les conteneurs qui sont spécifiés dans votre définition de tâche sont de type `NORMAL`. Vous pouvez ignorer les autres types de conteneurs, qui sont utilisés pour l'approvisionnement des ressources de tâches internes par l'agent de conteneur Amazon ECS.

`LogDriver`

Pilote de journal utilisé par le conteneur.

> **Note**
>
> Ces métadonnées `LogDriver` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure.

`LogOptions`

Options de pilote de journal définies pour le conteneur.

> **Note**
>
> Ces métadonnées `LogOptions` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure.

`ContainerARN`

L'Amazon Resource Name (ARN) du conteneur.

> **Note**
>
> Ces métadonnées `ContainerARN` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure.

`Networks`

Informations de réseau du conteneur, par exemple le mode réseau et l'adresse IP. Ce paramètre n'est pas spécifié si aucune information de réseau n'est définie.

`RestartCount`

Le nombre de fois que le conteneur a été redémarré.

> **Note**
>
> Les `RestartCount` métadonnées ne sont incluses que si une politique de redémarrage est activée pour le conteneur. Pour de plus amples informations, veuillez consulter Redémarrer des conteneurs individuels dans des tâches Amazon ECS avec des politiques de redémarrage de conteneurs.

`ExecutionStoppedAt`

Horodatage au moment où l'état des tâches est passé de `DesiredStatus` à `STOPPED`. Cela se produit lorsqu'un conteneur essentiel passe à `STOPPED`.

Exemples de métadonnées de tâches Amazon ECS v4

Les exemples suivants présentent des sorties de chaque point de terminaison des métadonnées de tâche.

Exemple de réponse de métadonnées du conteneur

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}`, seules les métadonnées sur le conteneur lui-même vous sont renvoyées. Voici un exemple de résultat d'une tâche exécutée dans le cadre d'un service (MyService).

```
{
    "DockerId": "ea32192c8553fbff06c9340478a2ff089b2bb5646fb718b4ee206641c9086d66",
    "Name": "curl",
    "DockerName": "ecs-curltest-24-curl-cca48e8dcadd97805600",
    "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
    "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
    "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/8f03e41243824aea923aca126495f665",
        "com.amazonaws.ecs.task-definition-family": "curltest",
```

```
        "com.amazonaws.ecs.task-definition-version": "24"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 10,
        "Memory": 128
    },
    "CreatedAt": "2020-10-02T00:15:07.620912337Z",
    "StartedAt": "2020-10-02T00:15:08.062559351Z",
    "Type": "NORMAL",
    "LogDriver": "awslogs",
    "LogOptions": {
        "awslogs-create-group": "true",
        "awslogs-group": "/ecs/metadata",
        "awslogs-region": "us-west-2",
        "awslogs-stream": "ecs/curl/8f03e41243824aea923aca126495f665"
    },
    "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/0206b271-
b33f-47ab-86c6-a0ba208a70a9",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "10.0.2.100"
            ],
            "AttachmentIndex": 0,
            "MACAddress": "0e:9e:32:c7:48:85",
            "IPv4SubnetCIDRBlock": "10.0.2.0/24",
            "PrivateDNSName": "ip-10-0-2-100.us-west-2.compute.internal",
            "SubnetGatewayIpv4Address": "10.0.2.1/24"
        }
    ]
}
```

Exemple de réponse de métadonnées de tâches

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/task`, les métadonnées relatives à la tâche dont le conteneur fait partie en plus des métadonnées pour chaque conteneur dans la tâche vous sont renvoyées. Voici un exemple de sortie.

```
{
    "Cluster": "default",
```

```
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
```

```
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
```

```
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Exemple de tâche avec réponse de métadonnées de balises

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`, les métadonnées relatives à la tâche, y compris les balises de tâche et d'instance de conteneur vous sont renvoyées. Voici un exemple de sortie.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "TaskTags": {
        "tag-use": "task-metadata-endpoint-test"
    },
    "ContainerInstanceTags":{
        "tag_key":"tag_value"
    },
    "LaunchType": "EC2",
    "Containers": [
        {
```

```
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
```

```
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
```

```
}
```

Exemple de tâche avec balises contenant une réponse de métadonnées d'erreur

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`, les métadonnées relatives à la tâche, y compris les balises de tâche et d'instance de conteneur vous sont renvoyées. S'il y a une erreur lors de la récupération des données d'étiquette, l'erreur est renvoyée dans la réponse. Ce qui suit est un exemple de sortie lorsque le rôle IAM associé à l'instance de conteneur n'a pas l'autorisation `ecs:ListTagsForResource`.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "Errors": [
        {
            "ErrorField": "ContainerInstanceTags",
            "ErrorCode": "AccessDeniedException",
            "ErrorMessage": "User: arn:aws:sts::111122223333:assumed-
role/ecsInstanceRole/i-0744a608689EXAMPLE is not authorized to perform:
 ecs:ListTagsForResource on resource: arn:aws:ecs:us-west-2:111122223333:container-
instance/default/2dd1b186f39845a584488d2ef155c131",
            "StatusCode": 400,
            "RequestId": "cd597ef0-272b-4643-9bd2-1de469870fa6",
            "ResourceARN": "arn:aws:ecs:us-west-2:111122223333:container-instance/
default/2dd1b186f39845a584488d2ef155c131"
        },
        {
            "ErrorField": "TaskTags",
            "ErrorCode": "AccessDeniedException",
            "ErrorMessage": "User: arn:aws:sts::111122223333:assumed-
role/ecsInstanceRole/i-0744a608689EXAMPLE is not authorized to perform:
 ecs:ListTagsForResource on resource: arn:aws:ecs:us-west-2:111122223333:task/
default/9ef30e4b7aa44d0db562749cff4983f3",
```

```
            "StatusCode": 400,
            "RequestId": "862c5986-6cd2-4aa6-87cc-70be395531e1",
            "ResourceARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/9ef30e4b7aa44d0db562749cff4983f3"
        }
    ],
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
```

```
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
```

```
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Exemple de réponse des statistiques de conteneur

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/stats`, les métriques réseau pour conteneur sont renvoyées. Pour les tâches Amazon ECS qui utilisent le mode `bridge` réseau `awsvpc` ou le mode réseau hébergés sur EC2 des instances Amazon exécutant au moins une version `1.43.0` de l'agent de conteneur, des statistiques supplémentaires sur le débit du réseau seront incluses dans la réponse. Pour toutes les autres tâches, la réponse inclura uniquement les statistiques cumulatives du réseau.

Voici un exemple de sortie d'une tâche Amazon ECS sur Amazon EC2 qui utilise le mode `bridge` réseau.

```
{
    "read": "2020-10-02T00:51:13.410254284Z",
    "preread": "2020-10-02T00:51:12.406202398Z",
    "pids_stats": {
        "current": 3
    },
    "blkio_stats": {
        "io_service_bytes_recursive": [

        ],
        "io_serviced_recursive": [

        ],
        "io_queue_recursive": [

        ],
        "io_service_time_recursive": [

        ],
        "io_wait_time_recursive": [
```

```
        ],
        "io_merged_recursive": [

        ],
        "io_time_recursive": [

        ],
        "sectors_recursive": [

        ]
    },
    "num_procs": 0,
    "storage_stats": {

    },
    "cpu_stats": {
        "cpu_usage": {
            "total_usage": 360968065,
            "percpu_usage": [
                182359190,
                178608875
            ],
            "usage_in_kernelmode": 40000000,
            "usage_in_usermode": 290000000
        },
        "system_cpu_usage": 13939680000000,
        "online_cpus": 2,
        "throttling_data": {
            "periods": 0,
            "throttled_periods": 0,
            "throttled_time": 0
        }
    },
    "precpu_stats": {
        "cpu_usage": {
            "total_usage": 360968065,
            "percpu_usage": [
                182359190,
                178608875
            ],
            "usage_in_kernelmode": 40000000,
            "usage_in_usermode": 290000000
        },
        "system_cpu_usage": 13937670000000,
```

```
        "online_cpus": 2,
        "throttling_data": {
            "periods": 0,
            "throttled_periods": 0,
            "throttled_time": 0
        }
    },
    "memory_stats": {
        "usage": 1806336,
        "max_usage": 6299648,
        "stats": {
            "active_anon": 606208,
            "active_file": 0,
            "cache": 0,
            "dirty": 0,
            "hierarchical_memory_limit": 134217728,
            "hierarchical_memsw_limit": 268435456,
            "inactive_anon": 0,
            "inactive_file": 0,
            "mapped_file": 0,
            "pgfault": 4185,
            "pgmajfault": 0,
            "pgpgin": 2926,
            "pgpgout": 2778,
            "rss": 606208,
            "rss_huge": 0,
            "total_active_anon": 606208,
            "total_active_file": 0,
            "total_cache": 0,
            "total_dirty": 0,
            "total_inactive_anon": 0,
            "total_inactive_file": 0,
            "total_mapped_file": 0,
            "total_pgfault": 4185,
            "total_pgmajfault": 0,
            "total_pgpgin": 2926,
            "total_pgpgout": 2778,
            "total_rss": 606208,
            "total_rss_huge": 0,
            "total_unevictable": 0,
            "total_writeback": 0,
            "unevictable": 0,
            "writeback": 0
        },
```

```
        "limit": 134217728
    },
    "name": "/ecs-curltest-26-curl-c2e5f6e0cf91b0bead01",
    "id": "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af",
    "networks": {
        "eth0": {
            "rx_bytes": 84,
            "rx_packets": 2,
            "rx_errors": 0,
            "rx_dropped": 0,
            "tx_bytes": 84,
            "tx_packets": 2,
            "tx_errors": 0,
            "tx_dropped": 0
        }
    },
    "network_rate_stats": {
        "rx_bytes_per_sec": 0,
        "tx_bytes_per_sec": 0
    }
}
```

Exemple de réponse des statistiques de tâche

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/task/stats`, les métriques réseau dont le conteneur fait partie vous sont renvoyées. Voici un exemple de sortie.

```
{
    "01999f2e5c6cf4df3873f28950e6278813408f281c54778efec860d0caad4854": {
        "read": "2020-10-02T00:51:32.51467703Z",
        "preread": "2020-10-02T00:51:31.50860463Z",
        "pids_stats": {
            "current": 1
        },
        "blkio_stats": {
            "io_service_bytes_recursive": [

            ],
            "io_serviced_recursive": [

            ],
            "io_queue_recursive": [
```

```
            ],
            "io_service_time_recursive": [

            ],
            "io_wait_time_recursive": [

            ],
            "io_merged_recursive": [

            ],
            "io_time_recursive": [

            ],
            "sectors_recursive": [

            ]
        },
        "num_procs": 0,
        "storage_stats": {

        },
        "cpu_stats": {
            "cpu_usage": {
                "total_usage": 177232665,
                "percpu_usage": [
                    13376224,
                    163856441
                ],
                "usage_in_kernelmode": 0,
                "usage_in_usermode": 160000000
            },
            "system_cpu_usage": 13977820000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "precpu_stats": {
            "cpu_usage": {
                "total_usage": 177232665,
                "percpu_usage": [
```

```
                13376224,
                163856441
            ],
            "usage_in_kernelmode": 0,
            "usage_in_usermode": 160000000
        },
        "system_cpu_usage": 13975800000000,
        "online_cpus": 2,
        "throttling_data": {
            "periods": 0,
            "throttled_periods": 0,
            "throttled_time": 0
        }
    },
    "memory_stats": {
        "usage": 532480,
        "max_usage": 6279168,
        "stats": {
            "active_anon": 40960,
            "active_file": 0,
            "cache": 0,
            "dirty": 0,
            "hierarchical_memory_limit": 9223372036854771712,
            "hierarchical_memsw_limit": 9223372036854771712,
            "inactive_anon": 0,
            "inactive_file": 0,
            "mapped_file": 0,
            "pgfault": 2033,
            "pgmajfault": 0,
            "pgpgin": 1734,
            "pgpgout": 1724,
            "rss": 40960,
            "rss_huge": 0,
            "total_active_anon": 40960,
            "total_active_file": 0,
            "total_cache": 0,
            "total_dirty": 0,
            "total_inactive_anon": 0,
            "total_inactive_file": 0,
            "total_mapped_file": 0,
            "total_pgfault": 2033,
            "total_pgmajfault": 0,
            "total_pgpgin": 1734,
            "total_pgpgout": 1724,
```

```
                "total_rss": 40960,
                "total_rss_huge": 0,
                "total_unevictable": 0,
                "total_writeback": 0,
                "unevictable": 0,
                "writeback": 0
            },
            "limit": 4073377792
        },
        "name": "/ecs-curltest-26-internalecspause-a6bcc3dbadfacfe85300",
        "id": "01999f2e5c6cf4df3873f28950e6278813408f281c54778efec860d0caad4854",
        "networks": {
            "eth0": {
                "rx_bytes": 84,
                "rx_packets": 2,
                "rx_errors": 0,
                "rx_dropped": 0,
                "tx_bytes": 84,
                "tx_packets": 2,
                "tx_errors": 0,
                "tx_dropped": 0
            }
        },
        "network_rate_stats": {
            "rx_bytes_per_sec": 0,
            "tx_bytes_per_sec": 0
        }
    },
    "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af": {
        "read": "2020-10-02T00:51:32.512771349Z",
        "preread": "2020-10-02T00:51:31.510597736Z",
        "pids_stats": {
            "current": 3
        },
        "blkio_stats": {
            "io_service_bytes_recursive": [

            ],
            "io_serviced_recursive": [

            ],
            "io_queue_recursive": [

            ],
```

```
            "io_service_time_recursive": [

            ],
            "io_wait_time_recursive": [

            ],
            "io_merged_recursive": [

            ],
            "io_time_recursive": [

            ],
            "sectors_recursive": [

            ]
        },
        "num_procs": 0,
        "storage_stats": {

        },
        "cpu_stats": {
            "cpu_usage": {
                "total_usage": 379075681,
                "percpu_usage": [
                    191355275,
                    187720406
                ],
                "usage_in_kernelmode": 60000000,
                "usage_in_usermode": 310000000
            },
            "system_cpu_usage": 13977800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "precpu_stats": {
            "cpu_usage": {
                "total_usage": 378825197,
                "percpu_usage": [
                    191104791,
                    187720406
```

```
                ],
                "usage_in_kernelmode": 60000000,
                "usage_in_usermode": 310000000
            },
            "system_cpu_usage": 13975800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "memory_stats": {
            "usage": 1814528,
            "max_usage": 6299648,
            "stats": {
                "active_anon": 606208,
                "active_file": 0,
                "cache": 0,
                "dirty": 0,
                "hierarchical_memory_limit": 134217728,
                "hierarchical_memsw_limit": 268435456,
                "inactive_anon": 0,
                "inactive_file": 0,
                "mapped_file": 0,
                "pgfault": 5377,
                "pgmajfault": 0,
                "pgpgin": 3613,
                "pgpgout": 3465,
                "rss": 606208,
                "rss_huge": 0,
                "total_active_anon": 606208,
                "total_active_file": 0,
                "total_cache": 0,
                "total_dirty": 0,
                "total_inactive_anon": 0,
                "total_inactive_file": 0,
                "total_mapped_file": 0,
                "total_pgfault": 5377,
                "total_pgmajfault": 0,
                "total_pgpgin": 3613,
                "total_pgpgout": 3465,
                "total_rss": 606208,
                "total_rss_huge": 0,
```

```
                "total_unevictable": 0,
                "total_writeback": 0,
                "unevictable": 0,
                "writeback": 0
            },
            "limit": 134217728
        },
        "name": "/ecs-curltest-26-curl-c2e5f6e0cf91b0bead01",
        "id": "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af",
        "networks": {
            "eth0": {
                "rx_bytes": 84,
                "rx_packets": 2,
                "rx_errors": 0,
                "rx_dropped": 0,
                "tx_bytes": 84,
                "tx_packets": 2,
                "tx_errors": 0,
                "tx_dropped": 0
            }
        },
        "network_rate_stats": {
            "rx_bytes_per_sec": 0,
            "tx_bytes_per_sec": 0
        }
    }
}
```

Point de terminaison des métadonnées des tâches Amazon ECS, version 3

> **Important**
>
> Le point de terminaison des métadonnées de tâches version 3 n'est plus activement maintenu. Nous vous recommandons de mettre à jour le point de terminaison des métadonnées de tâches version 4 afin d'obtenir les dernières informations relatives au point de terminaison des métadonnées. Pour de plus amples informations, veuillez consulter the section called “Point de terminaison des métadonnées de tâches version 4”.
> Si vous utilisez des tâches Amazon ECS hébergées sur AWS Fargate, consultez la version 3 du point de terminaison des métadonnées des tâches Amazon ECS pour les tâches sur Fargate.

À partir de la version 1.21.0 de l'agent de conteneur Amazon ECS, l'agent injecte une variable d'environnement appelée `ECS_CONTAINER_METADATA_URI` dans chaque conteneur d'une tâche. Lorsque vous interrogez le point de terminaison des métadonnées de tâches version 3, diverses métadonnées de tâches et statistiques Docker sont disponibles pour les tâches. Pour les tâches qui utilisent le mode réseau `bridge`, les métriques réseau sont disponibles sont disponibles lors de l'interrogation des points de terminaison `/stats`.

La fonctionnalité de point de terminaison des métadonnées des tâches version 3 est activée par défaut pour les tâches qui utilisent le type de lancement Fargate sur la version v1.3.0 ou ultérieure de la plateforme et les tâches qui utilisent EC2 le type de lancement et sont lancées sur une infrastructure EC2 Amazon Linux exécutant au moins la version 1.21.0 de l'agent de conteneur Amazon ECS ou sur une infrastructure EC2 Amazon Windows exécutant au moins une version `1.54.0` de l'agent de conteneur Amazon ECS et utilisant le mode réseau. `awsvpc` Pour de plus amples informations, veuillez consulter Gestion des instances de conteneurs Linux Amazon ECS.

Vous pouvez ajouter la prise en charge de cette fonctionnalité sur des anciennes instances de conteneur en mettant à jour l'agent à l'aide de la dernière version. Pour de plus amples informations, veuillez consulter Mise à jour de l'agent de conteneur Amazon ECS.

> **Important**
>
> Pour les tâches qui utilisent le type de lancement Fargate et les versions de plateforme antérieures à v1.3.0, le point de terminaison des métadonnées de tâches version 2 est pris en charge. Pour de plus amples informations, veuillez consulter Point de terminaison des métadonnées des tâches Amazon ECS, version 2.

Chemins du point de terminaison des métadonnées des tâches version 3

Les points de terminaison des métadonnées de tâches suivants sont disponibles pour les conteneurs :

`${ECS_CONTAINER_METADATA_URI}`

Ce chemin renvoie des métadonnées JSON pour le conteneur.

`${ECS_CONTAINER_METADATA_URI}/task`

Ce chemin renvoie des métadonnées JSON pour la tâche, y compris une liste des conteneurs IDs et les noms de tous les conteneurs associés à la tâche. Pour plus d'informations sur la réponse

pour ce point de terminaison, consultez Réponse JSON aux métadonnées des tâches Amazon ECS v3.

`${ECS_CONTAINER_METADATA_URI}/taskWithTags`

Ce chemin renvoie les métadonnées de la tâche incluse dans le point de terminaison `/task` en plus des balises de tâche et d'instance de conteneur qui peuvent être récupérées à l'aide de l'API `ListTagsForResource`.

`${ECS_CONTAINER_METADATA_URI}/stats`

Ce chemin renvoie le JSON des statistiques Docker pour cet ID de conteneur Docker spécifique. Pour plus d'informations sur chacune des statistiques renvoyées, consultez la documentation ContainerStatsde l'API Docker.

`${ECS_CONTAINER_METADATA_URI}/task/stats`

Ce chemin renvoie le JSON des statistiques Docker pour tous les conteneurs associés à la tâche. Pour plus d'informations sur chacune des statistiques renvoyées, consultez la documentation ContainerStatsde l'API Docker.

Réponse JSON aux métadonnées des tâches Amazon ECS v3

Les informations suivantes sont renvoyées à partir de la réponse JSON du point de terminaison de métadonnées de la tâche (`${ECS_CONTAINER_METADATA_URI}/task`).

`Cluster`

Amazon Resource Name (ARN) ou nom abrégé du cluster Amazon ECS auquel la tâche appartient.

`TaskARN`

Le nom de ressource Amazon (ARN) de la tâche à laquelle appartient le conteneur.

`Family`

Famille de la définition de tâche Amazon ECS pour la tâche.

`Revision`

Révision de la définition de tâche Amazon ECS pour la tâche.

`DesiredStatus`

État souhaité pour la tâche à partir d'Amazon ECS.

`KnownStatus`

État connu pour la tâche à partir d'Amazon ECS.

`Limits`

Les limites de ressources spécifiées au niveau de la tâche, telles que le processeur (exprimé en vCPUs) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`PullStartedAt`

Horodatage au moment où la première extraction de l'image de conteneur a commencé.

`PullStoppedAt`

Horodatage au moment où la dernière extraction de l'image de conteneur s'est terminée.

`AvailabilityZone`

Zone de disponibilité dans laquelle se trouve la tâche.

> **Note**
>
> Les métadonnées de la zone de disponibilité sont uniquement disponibles pour les tâches Fargate qui utilisent la version 1.4 de la plateforme ou ultérieure (Linux) ou la version 1.0.0 de la plateforme ou ultérieure (Windows).

`Containers`

Liste de métadonnées de conteneur pour chaque conteneur associé à la tâche.

`DockerId`

ID Docker du conteneur.

`Name`

Nom du conteneur tel que spécifié dans la définition de tâche.

`DockerName`

Nom du conteneur fourni à Docker. L'agent de conteneur Amazon ECS génère un nom unique pour le conteneur afin d'éviter les conflits de noms lorsque plusieurs copies de la même définition de tâche sont exécutées sur une seule instance.

`Image`

Image à utiliser pour le conteneur.

`ImageID`

Le condensé SHA-256 du manifeste d'image. Il s'agit du condensé qui peut être utilisé pour extraire l'image en utilisant le format`repository-url/image@sha256:digest`.

`Ports`

Tous les ports exposés pour le conteneur. Ce paramètre n'est pas spécifié s'il n'y a pas de ports exposés.

`Labels`

Toutes les étiquettes appliquées au conteneur. Ce paramètre n'est pas spécifié si aucune étiquette n'est appliquée.

`DesiredStatus`

État souhaité pour le conteneur à partir d'Amazon ECS.

`KnownStatus`

État connu pour le conteneur à partir d'Amazon ECS.

`ExitCode`

Code de sortie pour le conteneur. Ce paramètre n'est pas spécifié si le conteneur n'est pas sorti.

`Limits`

Limites de ressource spécifiées au niveau du conteneur, par exemple, le processeur (exprimé en unités de processeur) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`CreatedAt`

Horodatage de création du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été créé.

`StartedAt`

Horodatage de démarrage du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été démarré.

`FinishedAt`

Horodatage d'arrêt du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été arrêté.

`Type`

Type du conteneur. Les conteneurs qui sont spécifiés dans votre définition de tâche sont de type `NORMAL`. Vous pouvez ignorer les autres types de conteneurs, qui sont utilisés pour l'approvisionnement des ressources de tâches internes par l'agent de conteneur Amazon ECS.

`Networks`

Informations de réseau du conteneur, par exemple le mode réseau et l'adresse IP. Ce paramètre n'est pas spécifié si aucune information de réseau n'est définie.

`ClockDrift`

Informations sur la différence entre l'heure de référence et l'heure du système. Cela s'applique au système d'exploitation Linux. Cette fonctionnalité utilise Amazon Time Sync Service pour mesurer la précision de l'horloge et fournir l'erreur d'horloge liée aux conteneurs. Pour plus d'informations, consultez Définir l'heure pour votre instance Linux dans le Guide de EC2 l'utilisateur Amazon pour les instances Linux.

`ReferenceTime`

La base de la précision de l'horloge. Amazon ECS utilise la norme mondiale du temps universel coordonné (UTC) via NTP, par exemple `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La mesure de l'erreur de l'horloge est exprimée par le décalage par rapport à l'UTC. Cette erreur correspond à la différence en millisecondes entre l'heure de référence et l'heure du système.

`ClockSynchronizationStatus`

Indique si la tentative de synchronisation la plus récente entre l'heure système et l'heure de référence a réussi.

Les valeurs valides sont `SYNCHRONIZED` et `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Horodatage au moment où l'état des tâches est passé de `DesiredStatus` à `STOPPED`. Cela se produit lorsqu'un conteneur essentiel passe à `STOPPED`.

Exemples de métadonnées de tâches Amazon ECS v3

Les exemples suivants présentent des sorties de points de terminaison des métadonnées de tâche.

Exemple de réponse aux métadonnées du conteneur

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI}`, seules les métadonnées sur le conteneur lui-même vous sont renvoyées. Voici un exemple de sortie.

```
{
    "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
    "Name": "nginx-curl",
    "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
    "Image": "nrdlngr/nginx-curl",
    "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
    "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 512,
        "Memory": 512
    },
    "CreatedAt": "2018-02-01T20:55:10.554941919Z",
    "StartedAt": "2018-02-01T20:55:11.064236631Z",
    "Type": "NORMAL",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "10.0.2.106"
            ]
        }
    ]
}
```

Exemple de réponse de métadonnées de tâches

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI}/task`, les métadonnées relatives à la tâche dont le conteneur fait partie vous sont renvoyées. Voici un exemple de sortie.

La réponse JSON suivante est pour une tâche comportant un seul conteneur.

```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
```

```
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
```

```
}
```

Point de terminaison des métadonnées des tâches Amazon ECS, version 2

> **Important**
>
> Le point de terminaison des métadonnées de tâches version 2 n'est plus activement maintenu. Nous vous recommandons de mettre à jour le point de terminaison des métadonnées de tâches version 4 afin d'obtenir les dernières informations relatives au point de terminaison des métadonnées. Pour de plus amples informations, veuillez consulter the section called “Point de terminaison des métadonnées de tâches version 4”.

À partir de la version 1.17.0 de l'agent de conteneur Amazon ECS, différentes métadonnées de tâche et statistiques Docker sont disponibles pour les tâches qui utilisent le mode réseau `awsvpc` à un point de terminaison HTTP fourni par l'agent de conteneur Amazon ECS.

Tous les conteneurs appartenant à des tâches lancées en mode `awsvpc` réseau reçoivent une IPv4 adresse locale comprise dans une plage d'adresses lien-local prédéfinie. Lorsqu'un conteneur interroge le point de terminaison des métadonnées, l'agent de conteneur Amazon ECS peut déterminer à quelles tâches le conteneur appartient en fonction de son adresse IP unique. Des métadonnées et des statistiques pour cette tâche sont alors renvoyées.

Activation des métadonnées de tâche

La fonctionnalité des métadonnées de tâches version 2 est activée par défaut pour les éléments suivants :

- Les tâches utilisant le type de lancement Fargate et la version de plateforme v1.1.0 ou version ultérieure. Pour de plus amples informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.
- Tâches utilisant le type de EC2 lancement qui utilisent également le mode `awsvpc` réseau et sont lancées sur une infrastructure Amazon EC2 Linux exécutant au moins la version 1.17.0 de l'agent de conteneur Amazon ECS ou sur une infrastructure Amazon EC2 Windows exécutant au moins une version `1.54.0` de l'agent de conteneur Amazon ECS. Pour de plus amples informations, veuillez consulter Gestion des instances de conteneurs Linux Amazon ECS.

Vous pouvez ajouter la prise en charge de cette fonctionnalité sur des anciennes instances de conteneur en mettant à jour l'agent à l'aide de la dernière version. Pour de plus amples informations, veuillez consulter Mise à jour de l'agent de conteneur Amazon ECS.

Chemins de points de terminaison des métadonnées de tâches

Les points de terminaison d'API suivants sont disponibles pour les conteneurs :

`169.254.170.2/v2/metadata`

Ce point de terminaison renvoie des métadonnées JSON pour la tâche, y compris une liste des conteneurs IDs et les noms de tous les conteneurs associés à la tâche. Pour plus d'informations sur la réponse pour ce point de terminaison, consultez Réponse JSON des métadonnées de tâche.

`169.254.170.2/v2/metadata/<container-id>`

Ce point de terminaison renvoie le JSON des métadonnées pour l'ID de conteneur Docker spécifié.

`169.254.170.2/v2/metadata/taskWithTags`

Ce chemin renvoie les métadonnées de la tâche incluse dans le point de terminaison `/task` en plus des balises de tâche et d'instance de conteneur qui peuvent être récupérées à l'aide de l'API `ListTagsForResource`.

`169.254.170.2/v2/stats`

Ce point de terminaison renvoie le JSON des statistiques Docker pour tous les conteneurs associés à la tâche. Pour plus d'informations sur chacune des statistiques renvoyées, consultez la documentation ContainerStatsde l'API Docker.

`169.254.170.2/v2/stats/<container-id>`

Ce point de terminaison renvoie le JSON des statistiques Docker pour l'ID de conteneur Docker spécifié. Pour plus d'informations sur chacune des statistiques renvoyées, consultez la documentation ContainerStatsde l'API Docker.

Réponse JSON des métadonnées de tâche

Les informations suivantes sont renvoyées à partir de la réponse JSON du point de terminaison de métadonnées de la tâche (`169.254.170.2/v2/metadata`).

`Cluster`

Amazon Resource Name (ARN) ou nom abrégé du cluster Amazon ECS auquel la tâche appartient.

`TaskARN`

Le nom de ressource Amazon (ARN) de la tâche à laquelle appartient le conteneur.

`Family`

Famille de la définition de tâche Amazon ECS pour la tâche.

`Revision`

Révision de la définition de tâche Amazon ECS pour la tâche.

`DesiredStatus`

État souhaité pour la tâche à partir d'Amazon ECS.

`KnownStatus`

État connu pour la tâche à partir d'Amazon ECS.

`Limits`

Les limites de ressources spécifiées au niveau de la tâche, telles que le processeur (exprimé en vCPUs) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`PullStartedAt`

Horodatage au moment où la première extraction de l'image de conteneur a commencé.

`PullStoppedAt`

Horodatage au moment où la dernière extraction de l'image de conteneur s'est terminée.

`AvailabilityZone`

Zone de disponibilité dans laquelle se trouve la tâche.

> **Note**
>
> Les métadonnées de la zone de disponibilité sont uniquement disponibles pour les tâches Fargate qui utilisent la version 1.4 de la plateforme ou ultérieure (Linux) ou la version 1.0.0 de la plateforme ou ultérieure (Windows).

`Containers`

Liste de métadonnées de conteneur pour chaque conteneur associé à la tâche.

`DockerId`

ID Docker du conteneur.

`Name`

Nom du conteneur tel que spécifié dans la définition de tâche.

`DockerName`

Nom du conteneur fourni à Docker. L'agent de conteneur Amazon ECS génère un nom unique pour le conteneur afin d'éviter les conflits de noms lorsque plusieurs copies de la même définition de tâche sont exécutées sur une seule instance.

`Image`

Image à utiliser pour le conteneur.

`ImageID`

Le condensé SHA-256 du manifeste d'image. Il s'agit du condensé qui peut être utilisé pour extraire l'image en utilisant le format`repository-url/image@sha256:digest`.

`Ports`

Tous les ports exposés pour le conteneur. Ce paramètre n'est pas spécifié s'il n'y a pas de ports exposés.

`Labels`

Toutes les étiquettes appliquées au conteneur. Ce paramètre n'est pas spécifié si aucune étiquette n'est appliquée.

`DesiredStatus`

État souhaité pour le conteneur à partir d'Amazon ECS.

`KnownStatus`

État connu pour le conteneur à partir d'Amazon ECS.

`ExitCode`

Code de sortie pour le conteneur. Ce paramètre n'est pas spécifié si le conteneur n'est pas sorti.

`Limits`

Limites de ressource spécifiées au niveau du conteneur, par exemple, le processeur (exprimé en unités de processeur) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`CreatedAt`

Horodatage de création du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été créé.

`StartedAt`

Horodatage de démarrage du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été démarré.

`FinishedAt`

Horodatage d'arrêt du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été arrêté.

`Type`

Type du conteneur. Les conteneurs qui sont spécifiés dans votre définition de tâche sont de type `NORMAL`. Vous pouvez ignorer les autres types de conteneurs, qui sont utilisés pour l'approvisionnement des ressources de tâches internes par l'agent de conteneur Amazon ECS.

`Networks`

Informations de réseau du conteneur, par exemple le mode réseau et l'adresse IP. Ce paramètre n'est pas spécifié si aucune information de réseau n'est définie.

`ClockDrift`

Informations sur la différence entre l'heure de référence et l'heure du système. Cela s'applique au système d'exploitation Linux. Cette fonctionnalité utilise Amazon Time Sync Service pour mesurer la précision de l'horloge et fournir l'erreur d'horloge liée aux conteneurs. Pour plus d'informations, consultez Définir l'heure pour votre instance Linux dans le Guide de EC2 l'utilisateur Amazon pour les instances Linux.

`ReferenceTime`

La base de la précision de l'horloge. Amazon ECS utilise la norme mondiale du temps universel coordonné (UTC) via NTP, par exemple `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La mesure de l'erreur de l'horloge est exprimée par le décalage par rapport à l'UTC. Cette erreur correspond à la différence en millisecondes entre l'heure de référence et l'heure du système.

`ClockSynchronizationStatus`

Indique si la tentative de synchronisation la plus récente entre l'heure système et l'heure de référence a réussi.

Les valeurs valides sont `SYNCHRONIZED` et `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Horodatage au moment où l'état des tâches est passé de `DesiredStatus` à `STOPPED`. Cela se produit lorsqu'un conteneur essentiel passe à `STOPPED`.

Exemple de réponse de métadonnées de tâches

La réponse JSON suivante est pour une tâche comportant un seul conteneur.

```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
```

```
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
```

```
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
}
```

Métadonnées des tâches Amazon ECS disponibles pour les tâches sur Fargate

Amazon ECS sur Fargate fournit une méthode pour récupérer diverses métadonnées, métriques réseau et statistiques Docker relatives à vos conteneurs et aux tâches dont ils font partie. On parle alors du point de terminaison des métadonnées de tâches. Les versions de point de terminaison de métadonnées de tâche suivantes sont disponibles pour les tâches Amazon ECS sur Fargate :

- Point de terminaison des métadonnées de tâches version 4 – Disponible pour les tâches qui utilisent la version de plateforme 1.4.0 ou ultérieure.
- Point de terminaison des métadonnées de tâches version 3 – Disponible pour les tâches qui utilisent la version de plateforme 1.1.0 ou ultérieure.

Tous les conteneurs appartenant à des tâches lancées en mode `awsvpc` réseau reçoivent une IPv4 adresse locale comprise dans une plage d'adresses lien-local prédéfinie. Lorsqu'un conteneur interroge le point de terminaison des métadonnées, l'agent de conteneur peut déterminer à quelles tâches le conteneur appartient en fonction de son adresse IP unique. Des métadonnées et des statistiques pour cette tâche sont alors renvoyées.

Rubriques

- Point de terminaison de métadonnées de tâches Amazon ECS version 4 pour les tâches sur Fargate
- Point de terminaison de métadonnées de tâches Amazon ECS version 3 pour les tâches sur Fargate

Point de terminaison de métadonnées de tâches Amazon ECS version 4 pour les tâches sur Fargate

> **Important**
>
> Si vous utilisez des tâches Amazon ECS hébergées sur des EC2 instances Amazon, consultez le point de terminaison des métadonnées des tâches Amazon ECS.

À partir de la version `1.4.0` de la plateforme Fargate, une variable d'environnement nommée `ECS_CONTAINER_METADATA_URI_V4` est injectée dans chaque conteneur d'une tâche. Lorsque vous interrogez le point de terminaison des métadonnées de tâches version 4, diverses métadonnées de tâches et statistiques Docker sont disponibles pour les tâches.

Le point de terminaison des métadonnées de tâche version 4 fonctionne comme dans la version 3, mais fournit des métadonnées réseau supplémentaires pour vos conteneurs et tâches. Des métriques réseau supplémentaires sont disponibles lors de l'interrogation des points de terminaison `/stats`.

Le point de terminaison des métadonnées des tâches est activé par défaut pour toutes les tâches Amazon ECS exécutées sur AWS Fargate une version de plate-forme `1.4.0` ou ultérieure.

> **Note**
>
> Pour éviter de créer d'autres versions de point de terminaison de métadonnées de tâche à l'avenir, des métadonnées supplémentaires peuvent être ajoutées à la sortie de la version 4. Nous ne supprimons pas les métadonnées existantes et ne modifions pas les noms des champs de métadonnées.

Chemins des points de terminaison des métadonnées de tâches Fargate version 4

Les points de terminaison des métadonnées de tâches suivants sont disponibles pour les conteneurs :

`${ECS_CONTAINER_METADATA_URI_V4}`

Ce chemin renvoie des métadonnées pour le conteneur.

`${ECS_CONTAINER_METADATA_URI_V4}/task`

Ce chemin renvoie les métadonnées de la tâche, y compris une liste des conteneurs IDs et les noms de tous les conteneurs associés à la tâche. Pour plus d'informations sur la réponse pour ce point de terminaison, consultez Métadonnées de tâche Amazon ECS v4 : réponse JSON pour les tâches sur Fargate.

`${ECS_CONTAINER_METADATA_URI_V4}/stats`

Ce chemin renvoie les statistiques Docker pour le conteneur Docker. Pour plus d'informations sur chacune des statistiques renvoyées, consultez la documentation ContainerStatsde l'API Docker.

> **Note**
>
> Les tâches Amazon ECS suivantes AWS Fargate nécessitent que le conteneur soit exécuté pendant environ 1 seconde avant de renvoyer les statistiques du conteneur.

`${ECS_CONTAINER_METADATA_URI_V4}/task/stats`

Ce chemin renvoie les statistiques Docker pour tous les conteneurs associés à la tâche. Pour plus d'informations sur chacune des statistiques renvoyées, consultez la documentation ContainerStatsde l'API Docker.

> **Note**
>
> Les tâches Amazon ECS suivantes AWS Fargate nécessitent que le conteneur soit exécuté pendant environ 1 seconde avant de renvoyer les statistiques du conteneur.

Métadonnées de tâche Amazon ECS v4 : réponse JSON pour les tâches sur Fargate

Les métadonnées suivantes sont renvoyées dans la réponse JSON du point de terminaison de métadonnées de la tâche (`${ECS_CONTAINER_METADATA_URI_V4}/task`).

`Cluster`

Amazon Resource Name (ARN) ou nom abrégé du cluster Amazon ECS auquel la tâche appartient.

`VPCID`

L'ID VPC de l'instance de EC2 conteneur Amazon. Ce champ n'apparaît que pour les EC2 instances Amazon.

> **Note**
>
> Les métadonnées `VPCID` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS `1.63.1` ou version ultérieure.

`TaskARN`

Le nom de ressource Amazon (ARN) de la tâche à laquelle appartient le conteneur.

`Family`

Famille de la définition de tâche Amazon ECS pour la tâche.

`Revision`

Révision de la définition de tâche Amazon ECS pour la tâche.

`DesiredStatus`

État souhaité pour la tâche à partir d'Amazon ECS.

`KnownStatus`

État connu pour la tâche à partir d'Amazon ECS.

`Limits`

Les limites de ressources spécifiées au niveau des tâches, telles que le processeur (exprimé en vCPUs) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`PullStartedAt`

Horodatage au moment où la première extraction de l'image de conteneur a commencé.

`PullStoppedAt`

Horodatage au moment où la dernière extraction de l'image de conteneur s'est terminée.

`AvailabilityZone`

Zone de disponibilité dans laquelle se trouve la tâche.

> **Note**
>
> Les métadonnées de la zone de disponibilité sont uniquement disponibles pour les tâches Fargate qui utilisent la version 1.4 de la plateforme ou ultérieure (Linux) ou la version 1.0.0 de la plateforme (Windows).

`LaunchType`

Type de lancement utilisé par la tâche. Lorsque vous utilisez des fournisseurs de capacité de cluster, cela indique si la tâche utilise EC2 Fargate ou une infrastructure.

> **Note**
>
> Ces métadonnées `LaunchType` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure (Linux) ou 1.0.0 ou ultérieure (Windows).

`EphemeralStorageMetrics`

La taille réservée et l'utilisation actuelle du stockage éphémère de cette tâche.

> **Note**
>
> Fargate réserve de l'espace sur le disque. Il n'est utilisé que par Fargate. Vous n'êtes pas facturé pour cela. Cela n'apparaît pas dans ces statistiques. Toutefois, vous pouvez voir ce stockage supplémentaire dans d'autres outils tels que `df`.

`Utilized`

La quantité de stockage éphémère actuellement utilisée (en Mio) pour cette tâche.

`Reserved`

La quantité de stockage éphémère réservée (en Mio) pour cette tâche. La taille de stockage éphémère ne peut pas être modifiée dans une tâche en cours d'exécution. Vous pouvez spécifier l'objet du `ephermalStorage` dans la définition de votre tâche pour modifier la

quantité de stockage éphémère. Le `ephermalStorage` est spécifié en Gio, et non en Mio. Le `ephermalStorage` et les `EphemeralStorageMetrics` sont uniquement disponibles sur la plateforme Fargate Linux version 1.4.0 ou ultérieure.

`Containers`

Liste de métadonnées de conteneur pour chaque conteneur associé à la tâche.

`DockerId`

ID Docker du conteneur.

Lorsque vous utilisez Fargate, l'ID est un hexadécimal de 32 chiffres suivi d'un nombre de 10 chiffres.

`Name`

Nom du conteneur tel que spécifié dans la définition de tâche.

`DockerName`

Nom du conteneur fourni à Docker. L'agent de conteneur Amazon ECS génère un nom unique pour le conteneur afin d'éviter les conflits de noms lorsque plusieurs copies de la même définition de tâche sont exécutées sur une seule instance.

`Image`

Image à utiliser pour le conteneur.

`ImageID`

Le condensé SHA-256 du manifeste d'image. Il s'agit du condensé qui peut être utilisé pour extraire l'image en utilisant le format`repository-url/image@sha256:digest`.

`Ports`

Tous les ports exposés pour le conteneur. Ce paramètre n'est pas spécifié s'il n'y a pas de ports exposés.

`Labels`

Toutes les étiquettes appliquées au conteneur. Ce paramètre n'est pas spécifié si aucune étiquette n'est appliquée.

`DesiredStatus`

État souhaité pour le conteneur à partir d'Amazon ECS.

`KnownStatus`

État connu pour le conteneur à partir d'Amazon ECS.

`ExitCode`

Code de sortie pour le conteneur. Ce paramètre n'est pas spécifié si le conteneur n'est pas sorti.

`Limits`

Limites de ressource spécifiées au niveau du conteneur, par exemple, le processeur (exprimé en unités de processeur) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`CreatedAt`

Horodatage de création du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été créé.

`StartedAt`

Horodatage de démarrage du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été démarré.

`FinishedAt`

Horodatage d'arrêt du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été arrêté.

`Type`

Type du conteneur. Les conteneurs qui sont spécifiés dans votre définition de tâche sont de type `NORMAL`. Vous pouvez ignorer les autres types de conteneurs, qui sont utilisés pour l'approvisionnement des ressources de tâches internes par l'agent de conteneur Amazon ECS.

`LogDriver`

Pilote de journal utilisé par le conteneur.

> **Note**
>
> Ces métadonnées `LogDriver` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure.

`LogOptions`

Options de pilote de journal définies pour le conteneur.

> **Note**
>
> Ces métadonnées `LogOptions` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure.

`ContainerARN`

L'Amazon Resource Name (ARN) du conteneur.

> **Note**
>
> Ces métadonnées `ContainerARN` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure.

`Networks`

Informations de réseau du conteneur, par exemple le mode réseau et l'adresse IP. Ce paramètre n'est pas spécifié si aucune information de réseau n'est définie.

`Snapshotter`

Le snapshotter qui a été utilisé par containerd pour télécharger cette image de conteneur. Les valeurs valides sont `overlayfs`, qui est la valeur par défaut, et `soci`, utilisées en cas de chargement différé avec un index SOCI. Ce paramètre est uniquement disponible pour les tâches qui s'exécutent sur une version de plateforme Linux `1.4.0`.

`RestartCount`

Le nombre de fois que le conteneur a été redémarré.

> **Note**
>
> Les `RestartCount` métadonnées ne sont incluses que si une politique de redémarrage est activée pour le conteneur. Pour de plus amples informations, veuillez

consulter Redémarrer des conteneurs individuels dans des tâches Amazon ECS avec des politiques de redémarrage de conteneurs.

`ClockDrift`

Informations sur la différence entre l'heure de référence et l'heure du système. Cette fonctionnalité utilise Amazon Time Sync Service pour mesurer la précision de l'horloge et fournir l'erreur d'horloge liée aux conteneurs. Pour plus d'informations, consultez Définir l'heure pour votre instance Linux dans le Guide de EC2 l'utilisateur Amazon pour les instances Linux.

`ReferenceTime`

La base de la précision de l'horloge. Amazon ECS utilise la norme mondiale du temps universel coordonné (UTC) via NTP, par exemple `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La mesure de l'erreur de l'horloge est exprimée par le décalage par rapport à l'UTC. Cette erreur correspond à la différence en millisecondes entre l'heure de référence et l'heure du système.

`ClockSynchronizationStatus`

Indique si la tentative de synchronisation la plus récente entre l'heure système et l'heure de référence a réussi.

Les valeurs valides sont `SYNCHRONIZED` et `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Horodatage au moment où l'état des tâches est passé de `DesiredStatus` à `STOPPED`. Cela se produit lorsqu'un conteneur essentiel passe à `STOPPED`.

Exemples de métadonnées de tâches Amazon ECS v4 pour les tâches sur Fargate

Les exemples suivants présentent des sorties de points de terminaison des métadonnées de tâche pour les tâches Amazon ECS exécutées sur AWS Fargate.

Depuis le conteneur, vous pouvez utiliser curl suivi du point de terminaison des métadonnées de la tâche pour interroger le point de terminaison, par exemple `curl ${ECS_CONTAINER_METADATA_URI_V4}/task`.

Exemple de réponse de métadonnées du conteneur

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}`, seules les métadonnées sur le conteneur lui-même vous sont renvoyées. Voici un exemple de sortie.

```
{
    "DockerId": "cd189a933e5849daa93386466019ab50-2495160603",
    "Name": "curl",
    "DockerName": "curl",
    "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
    "ImageID":
 "sha256:25f3695bedfb454a50f12d127839a68ad3caf91e451c1da073db34c542c4d2cb",
    "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-west-2:111122223333:cluster/
default",
        "com.amazonaws.ecs.container-name": "curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/default/
cd189a933e5849daa93386466019ab50",
        "com.amazonaws.ecs.task-definition-family": "curltest",
        "com.amazonaws.ecs.task-definition-version": "2"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 10,
        "Memory": 128
    },
    "CreatedAt": "2020-10-08T20:09:11.44527186Z",
    "StartedAt": "2020-10-08T20:09:11.44527186Z",
    "Type": "NORMAL",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "192.0.2.3"
            ],
            "AttachmentIndex": 0,
            "MACAddress": "0a:de:f6:10:51:e5",
            "IPv4SubnetCIDRBlock": "192.0.2.0/24",
            "DomainNameServers": [
                "192.0.2.2"
            ],
            "DomainNameSearchList": [
                "us-west-2.compute.internal"
```

```
            ],
            "PrivateDNSName": "ip-10-0-0-222.us-west-2.compute.internal",
            "SubnetGatewayIpv4Address": "192.0.2.0/24"
        }
    ],
    "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/05966557-
f16c-49cb-9352-24b3a0dcd0e1",
    "LogOptions": {
        "awslogs-create-group": "true",
        "awslogs-group": "/ecs/containerlogs",
        "awslogs-region": "us-west-2",
        "awslogs-stream": "ecs/curl/cd189a933e5849daa93386466019ab50"
    },
    "LogDriver": "awslogs",
    "Snapshotter": "overlayfs"
}
```

Exemples de métadonnées de tâches Amazon ECS v4 pour les tâches sur Fargate

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/task`, les métadonnées relatives à la tâche dont le conteneur fait partie vous sont renvoyées. Voici un exemple de sortie.

```
{
  "Cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/MyEmptyCluster",
  "TaskARN": "arn:aws:ecs:us-east-1:123456789012:task/MyEmptyCluster/
bfa2636268144d039771334145e490c5",
  "Family": "sample-fargate",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Limits": {
    "CPU": 0.25,
    "Memory": 512
  },
  "PullStartedAt": "2023-07-21T15:45:33.532811081Z",
  "PullStoppedAt": "2023-07-21T15:45:38.541068435Z",
  "AvailabilityZone": "us-east-1d",
  "Containers": [
    {
      "DockerId": "bfa2636268144d039771334145e490c5-1117626119",
      "Name": "curl-image",
      "DockerName": "curl-image",
```

```
      "Image": "curlimages/curl",
      "ImageID":
 "sha256:daf3f46a2639c1613b25e85c9ee4193af8a1d538f92483d67f9a3d7f21721827",
      "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/
MyEmptyCluster",
        "com.amazonaws.ecs.container-name": "curl-image",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-east-1:123456789012:task/
MyEmptyCluster/bfa2636268144d039771334145e490c5",
        "com.amazonaws.ecs.task-definition-family": "sample-fargate",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": { "CPU": 128 },
      "CreatedAt": "2023-07-21T15:45:44.91368314Z",
      "StartedAt": "2023-07-21T15:45:44.91368314Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": ["172.31.42.189"],
          "AttachmentIndex": 0,
          "MACAddress": "0e:98:9f:33:76:d3",
          "IPv4SubnetCIDRBlock": "172.31.32.0/20",
          "DomainNameServers": ["172.31.0.2"],
          "DomainNameSearchList": ["ec2.internal"],
          "PrivateDNSName": "ip-172-31-42-189.ec2.internal",
          "SubnetGatewayIpv4Address": "172.31.32.1/20"
        }
      ],
      "ContainerARN": "arn:aws:ecs:us-east-1:123456789012:container/MyEmptyCluster/
bfa2636268144d039771334145e490c5/da6cccf7-1178-400c-afdf-7536173ee209",
      "Snapshotter": "overlayfs"
    },
    {
      "DockerId": "bfa2636268144d039771334145e490c5-3681984407",
      "Name": "fargate-app",
      "DockerName": "fargate-app",
      "Image": "public.ecr.aws/docker/library/httpd:latest",
      "ImageID":
 "sha256:8059bdd0058510c03ae4c808de8c4fd2c1f3c1b6d9ea75487f1e5caa5ececa02",
      "Labels": {
```

```
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/
MyEmptyCluster",
        "com.amazonaws.ecs.container-name": "fargate-app",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-east-1:123456789012:task/
MyEmptyCluster/bfa2636268144d039771334145e490c5",
        "com.amazonaws.ecs.task-definition-family": "sample-fargate",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": { "CPU": 2 },
      "CreatedAt": "2023-07-21T15:45:44.954460255Z",
      "StartedAt": "2023-07-21T15:45:44.954460255Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": ["172.31.42.189"],
          "AttachmentIndex": 0,
          "MACAddress": "0e:98:9f:33:76:d3",
          "IPv4SubnetCIDRBlock": "172.31.32.0/20",
          "DomainNameServers": ["172.31.0.2"],
          "DomainNameSearchList": ["ec2.internal"],
          "PrivateDNSName": "ip-172-31-42-189.ec2.internal",
          "SubnetGatewayIpv4Address": "172.31.32.1/20"
        }
      ],
      "ContainerARN": "arn:aws:ecs:us-east-1:123456789012:container/MyEmptyCluster/
bfa2636268144d039771334145e490c5/f65b461d-aa09-4acb-a579-9785c0530cbc",
      "Snapshotter": "overlayfs"
    }
  ],
  "LaunchType": "FARGATE",
  "ClockDrift": {
    "ClockErrorBound": 0.446931,
    "ReferenceTimestamp": "2023-07-21T16:09:17Z",
    "ClockSynchronizationStatus": "SYNCHRONIZED"
  },
  "EphemeralStorageMetrics": {
    "Utilized": 261,
    "Reserved": 20496
  }
}
```

Exemple de réponse des statistiques de tâche

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/task/stats`, les métriques réseau dont le conteneur fait partie vous sont renvoyées. Voici un exemple de sortie.

```
{
  "3d1f891cded94dc795608466cce8ddcf-464223573": {
    "read": "2020-10-08T21:24:44.938937019Z",
    "preread": "2020-10-08T21:24:34.938633969Z",
    "pids_stats": {},
    "blkio_stats": {
      "io_service_bytes_recursive": [
        {
          "major": 202,
          "minor": 26368,
          "op": "Read",
          "value": 638976
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Write",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Sync",
          "value": 638976
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Async",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Total",
          "value": 638976
        }
```

```
      ],
      "io_serviced_recursive": [
        {
          "major": 202,
          "minor": 26368,
          "op": "Read",
          "value": 12
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Write",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Sync",
          "value": 12
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Async",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Total",
          "value": 12
        }
      ],
      "io_queue_recursive": [],
      "io_service_time_recursive": [],
      "io_wait_time_recursive": [],
      "io_merged_recursive": [],
      "io_time_recursive": [],
      "sectors_recursive": []
    },
    "num_procs": 0,
    "storage_stats": {},
    "cpu_stats": {
      "cpu_usage": {
```

```
        "total_usage": 1137691504,
        "percpu_usage": [
          696479228,
          441212276,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0
        ],
        "usage_in_kernelmode": 80000000,
        "usage_in_usermode": 810000000
      },
      "system_cpu_usage": 9393210000000,
      "online_cpus": 2,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
        "throttled_time": 0
      }
    },
    "precpu_stats": {
      "cpu_usage": {
        "total_usage": 1136624601,
        "percpu_usage": [
          695639662,
          440984939,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
```

```
          0,
          0,
          0,
          0
        ],
        "usage_in_kernelmode": 80000000,
        "usage_in_usermode": 810000000
      },
      "system_cpu_usage": 9373330000000,
      "online_cpus": 2,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
        "throttled_time": 0
      }
    },
    "memory_stats": {
      "usage": 6504448,
      "max_usage": 8458240,
      "stats": {
        "active_anon": 1675264,
        "active_file": 557056,
        "cache": 651264,
        "dirty": 0,
        "hierarchical_memory_limit": 536870912,
        "hierarchical_memsw_limit": 9223372036854772000,
        "inactive_anon": 0,
        "inactive_file": 3088384,
        "mapped_file": 430080,
        "pgfault": 11034,
        "pgmajfault": 5,
        "pgpgin": 8436,
        "pgpgout": 7137,
        "rss": 4669440,
        "rss_huge": 0,
        "total_active_anon": 1675264,
        "total_active_file": 557056,
        "total_cache": 651264,
        "total_dirty": 0,
        "total_inactive_anon": 0,
        "total_inactive_file": 3088384,
        "total_mapped_file": 430080,
        "total_pgfault": 11034,
        "total_pgmajfault": 5,
```

```
        "total_pgpgin": 8436,
        "total_pgpgout": 7137,
        "total_rss": 4669440,
        "total_rss_huge": 0,
        "total_unevictable": 0,
        "total_writeback": 0,
        "unevictable": 0,
        "writeback": 0
      },
      "limit": 9223372036854772000
    },
    "name": "curltest",
    "id": "3d1f891cded94dc795608466cce8ddcf-464223573",
    "networks": {
      "eth1": {
        "rx_bytes": 2398415937,
        "rx_packets": 1898631,
        "rx_errors": 0,
        "rx_dropped": 0,
        "tx_bytes": 1259037719,
        "tx_packets": 428002,
        "tx_errors": 0,
        "tx_dropped": 0
      }
    },
    "network_rate_stats": {
      "rx_bytes_per_sec": 43.298687872232854,
      "tx_bytes_per_sec": 215.39347269466413
    }
  }
}
```

Point de terminaison de métadonnées de tâches Amazon ECS version 3 pour les tâches sur Fargate

> **Important**
>
> Le point de terminaison des métadonnées de tâches version 3 n'est plus activement maintenu. Nous vous recommandons de mettre à jour le point de terminaison des métadonnées de tâches version 4 afin d'obtenir les dernières informations relatives au point de terminaison des métadonnées. Pour de plus amples informations, veuillez consulter the

section called “Point de terminaison des métadonnées de tâches version 4 pour les tâches sur Fargate”.

À partir de la version `1.1.0` de la plateforme Fargate, une variable d'environnement nommée `ECS_CONTAINER_METADATA_URI` est injectée dans chaque conteneur d'une tâche. Lorsque vous interrogez le point de terminaison des métadonnées de tâches version 3, diverses métadonnées de tâches et statistiques Docker sont disponibles pour les tâches.

La fonction de point de terminaison des métadonnées de tâches est activée par défaut pour les tâches Amazon ECS hébergées sur Fargate qui utilisent la version de plateforme `1.1.0` ou ultérieure. Pour de plus amples informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.

Chemins de point de terminaison des métadonnées de tâches sur Fargate

Les points de terminaison d'API suivants sont disponibles pour les conteneurs :

`${ECS_CONTAINER_METADATA_URI}`

Ce chemin renvoie des métadonnées JSON pour le conteneur.

`${ECS_CONTAINER_METADATA_URI}/task`

Ce chemin renvoie des métadonnées JSON pour la tâche, y compris une liste des conteneurs IDs et les noms de tous les conteneurs associés à la tâche. Pour plus d'informations sur la réponse pour ce point de terminaison, consultez Métadonnées de tâche Amazon ECS v3 : réponse JSON pour les tâches sur Fargate.

`${ECS_CONTAINER_METADATA_URI}/stats`

Ce chemin renvoie le JSON des statistiques Docker pour cet ID de conteneur Docker spécifique. Pour plus d'informations sur chacune des statistiques renvoyées, consultez la documentation ContainerStatsde l'API Docker.

`${ECS_CONTAINER_METADATA_URI}/task/stats`

Ce chemin renvoie le JSON des statistiques Docker pour tous les conteneurs associés à la tâche. Pour plus d'informations sur chacune des statistiques renvoyées, consultez la documentation ContainerStatsde l'API Docker.

Métadonnées de tâche Amazon ECS v3 : réponse JSON pour les tâches sur Fargate

Les informations suivantes sont renvoyées à partir de la réponse JSON du point de terminaison de métadonnées de la tâche (`${ECS_CONTAINER_METADATA_URI}/task`).

`Cluster`

Amazon Resource Name (ARN) ou nom abrégé du cluster Amazon ECS auquel la tâche appartient.

`TaskARN`

Le nom de ressource Amazon (ARN) de la tâche à laquelle appartient le conteneur.

`Family`

Famille de la définition de tâche Amazon ECS pour la tâche.

`Revision`

Révision de la définition de tâche Amazon ECS pour la tâche.

`DesiredStatus`

État souhaité pour la tâche à partir d'Amazon ECS.

`KnownStatus`

État connu pour la tâche à partir d'Amazon ECS.

`Limits`

Les limites de ressources spécifiées au niveau de la tâche, telles que le processeur (exprimé en vCPUs) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`PullStartedAt`

Horodatage au moment où la première extraction de l'image de conteneur a commencé.

`PullStoppedAt`

Horodatage au moment où la dernière extraction de l'image de conteneur s'est terminée.

`AvailabilityZone`

Zone de disponibilité dans laquelle se trouve la tâche.

> **Note**
>
> Les métadonnées de la zone de disponibilité sont uniquement disponibles pour les tâches Fargate qui utilisent la version 1.4 de la plateforme ou ultérieure (Linux) ou la version 1.0.0 de la plateforme ou ultérieure (Windows).

`Containers`

Liste de métadonnées de conteneur pour chaque conteneur associé à la tâche.

`DockerId`

ID Docker du conteneur.

`Name`

Nom du conteneur tel que spécifié dans la définition de tâche.

`DockerName`

Nom du conteneur fourni à Docker. L'agent de conteneur Amazon ECS génère un nom unique pour le conteneur afin d'éviter les conflits de noms lorsque plusieurs copies de la même définition de tâche sont exécutées sur une seule instance.

`Image`

Image à utiliser pour le conteneur.

`ImageID`

Le condensé SHA-256 du manifeste d'image. Il s'agit du condensé qui peut être utilisé pour extraire l'image en utilisant le format`repository-url/image@sha256:digest`.

`Ports`

Tous les ports exposés pour le conteneur. Ce paramètre n'est pas spécifié s'il n'y a pas de ports exposés.

`Labels`

Toutes les étiquettes appliquées au conteneur. Ce paramètre n'est pas spécifié si aucune étiquette n'est appliquée.

`DesiredStatus`

État souhaité pour le conteneur à partir d'Amazon ECS.

`KnownStatus`

État connu pour le conteneur à partir d'Amazon ECS.

`ExitCode`

Code de sortie pour le conteneur. Ce paramètre n'est pas spécifié si le conteneur n'est pas sorti.

`Limits`

Limites de ressource spécifiées au niveau du conteneur, par exemple, le processeur (exprimé en unités de processeur) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`CreatedAt`

Horodatage de création du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été créé.

`StartedAt`

Horodatage de démarrage du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été démarré.

`FinishedAt`

Horodatage d'arrêt du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été arrêté.

`Type`

Type du conteneur. Les conteneurs qui sont spécifiés dans votre définition de tâche sont de type `NORMAL`. Vous pouvez ignorer les autres types de conteneurs, qui sont utilisés pour l'approvisionnement des ressources de tâches internes par l'agent de conteneur Amazon ECS.

`Networks`

Informations de réseau du conteneur, par exemple le mode réseau et l'adresse IP. Ce paramètre n'est pas spécifié si aucune information de réseau n'est définie.

`ClockDrift`

Informations sur la différence entre l'heure de référence et l'heure du système. Cela s'applique au système d'exploitation Linux. Cette fonctionnalité utilise Amazon Time Sync Service pour mesurer la précision de l'horloge et fournir l'erreur d'horloge liée aux conteneurs. Pour plus d'informations,

consultez Définir l'heure pour votre instance Linux dans le Guide de EC2 l'utilisateur Amazon pour les instances Linux.

`ReferenceTime`

La base de la précision de l'horloge. Amazon ECS utilise la norme mondiale du temps universel coordonné (UTC) via NTP, par exemple `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La mesure de l'erreur de l'horloge est exprimée par le décalage par rapport à l'UTC. Cette erreur correspond à la différence en millisecondes entre l'heure de référence et l'heure du système.

`ClockSynchronizationStatus`

Indique si la tentative de synchronisation la plus récente entre l'heure système et l'heure de référence a réussi.

Les valeurs valides sont `SYNCHRONIZED` et `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Horodatage au moment où l'état des tâches est passé de `DesiredStatus` à `STOPPED`. Cela se produit lorsqu'un conteneur essentiel passe à `STOPPED`.

Exemples de métadonnées de tâches Amazon ECS v3 pour les tâches sur Fargate

La réponse JSON suivante est pour une tâche comportant un seul conteneur.

```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
```

```
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
```

```
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
}
```

Introspection des conteneurs Amazon ECS

L'agent de conteneur Amazon ECS fournit une opération API pour l'obtention d'informations sur l'instance de conteneur sur laquelle l'agent est exécuté et sur les tâches connexes elles aussi exécutées sur cette instance. Vous pouvez utiliser la commande curl à partir de l'instance de conteneur pour interroger l'agent de conteneur Amazon ECS (port 51678) et renvoyer des métadonnées d'instance de conteneur ou des informations sur les tâches.

> **Important**
>
> Votre instance de conteneur doit avoir un rôle IAM qui autorise l'accès à Amazon ECS afin de récupérer les métadonnées. Pour de plus amples informations, veuillez consulter Rôle IAM d'instance de conteneur Amazon ECS.

Pour afficher les métadonnées de l'instance de conteneur, connectez-vous à votre instance de conteneur via SSH et exécutez la commande suivante. Les métadonnées comprennent l'ID de l'instance de conteneur, le cluster Amazon ECS dans lequel l'instance de conteneur est enregistrée, ainsi que les informations de version de l'agent de conteneur Amazon ECS.

```
curl -s http://localhost:51678/v1/metadata | python3 -mjson.tool
```

Sortie :

```
{
    "Cluster": "cluster_name",
    "ContainerInstanceArn": "arn:aws:ecs:region:aws_account_id:container-
instance/cluster_name/container_instance_id",
    "Version": "Amazon ECS Agent - v1.30.0 (02ff320c)"
}
```

Pour afficher les informations sur toutes les tâches qui sont exécutées sur une instance de conteneur, connectez-vous à votre instance de conteneur via SSH et exécutez la commande suivante :

```
curl http://localhost:51678/v1/tasks
```

Sortie :

```
{
    "Tasks": [
        {
            "Arn": "arn:aws:ecs:us-west-2:012345678910:task/default/example5-58ff-46c9-
ae05-543f8example",
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Family": "hello_world",
            "Version": "8",
            "Containers": [
                {
          "DockerId":
 "9581a69a761a557fbfce1d0f6745e4af5b9dbfb86b6b2c5c4df156f1a5932ff1",
          "DockerName": "ecs-hello_world-8-mysql-fcae8ac8f9f1d89d8301",
          "Name": "mysql",
          "CreatedAt": "2023-10-08T20:09:11.44527186Z",
          "StartedAt": "2023-10-08T20:09:11.44527186Z",
          "ImageID":
 "sha256:2ae34abc2ed0a22e280d17e13f9c01aaf725688b09b7a1525d1a2750e2c0d1de"
        },
        {
          "DockerId":
 "bf25c5c5b2d4dba68846c7236e75b6915e1e778d31611e3c6a06831e39814a15",
```

```
            "DockerName": "ecs-hello_world-8-wordpress-e8bfddf9b488dff36c00",
            "Name": "wordpress"
        }
    ]
}
    ]
}
```

Vous pouvez afficher les informations relatives à une tâche spécifique en cours d'exécution sur une instance de conteneur. Pour spécifier une tâche ou un conteneur spécifique, ajoutez un des éléments suivants à la demande :

- La tâche ARN (`?taskarn=`*`task_arn`*)
- L'ID Docker d'un conteneur (`?dockerid=`*`docker_id`*)

Pour obtenir des informations sur une tâche avec l'ID Docker d'un conteneur, connectez-vous à votre instance de conteneur via SSH et exécutez la commande suivante.

> **Note**
>
> Les agents de conteneur Amazon ECS antérieurs à la version 1.14.2 nécessitent un conteneur Docker complet IDs pour l'API d'introspection, et non la version courte présentée avec. docker ps Vous pouvez obtenir l'ID Docker complet d'un conteneur en exécutant la commande docker ps --no-trunc sur l'instance du conteneur.

```
curl http://localhost:51678/v1/tasks?dockerid=79c796ed2a7f
```

Sortie :

```
{
    "Arn": "arn:aws:ecs:us-west-2:012345678910:task/default/e01d58a8-151b-40e8-
bc01-22647b9ecfec",
    "Containers": [
        {
            "DockerId":
 "79c796ed2a7f864f485c76f83f3165488097279d296a7c05bd5201a1c69b2920",
            "DockerName": "ecs-nginx-efs-2-nginx-9ac0808dd0afa495f001",
```

```
            "Name": "nginx",
            "CreatedAt": "2023-10-08T20:09:11.44527186Z",
            "StartedAt": "2023-10-08T20:09:11.44527186Z",
            "ImageID":
 "sha256:2ae34abc2ed0a22e280d17e13f9c01aaf725688b09b7a1525d1a2750e2c0d1de"
        }
    ],
    "DesiredStatus": "RUNNING",
    "Family": "nginx-efs",
    "KnownStatus": "RUNNING",
    "Version": "2"
}
```

Identifiez les comportements non autorisés grâce à la surveillance du temps d'exécution

Amazon GuardDuty est un service de détection des menaces qui aide à protéger vos comptes, vos conteneurs, vos charges de travail et les données de votre AWS environnement. À l'aide de modèles d'apprentissage automatique (ML) et de capacités de détection des anomalies et des menaces, vous surveillez GuardDuty en permanence les différentes sources de journaux et l'activité d'exécution afin d'identifier et de hiérarchiser les risques de sécurité potentiels et les activités malveillantes dans votre environnement.

La surveillance du temps d'exécution GuardDuty protège les charges de travail exécutées sur Fargate EC2 et les instances de conteneur en AWS surveillant en permanence l'activité des journaux et du réseau afin d'identifier les comportements malveillants ou non autorisés. Runtime Monitoring utilise un agent de GuardDuty sécurité léger et entièrement géré qui analyse le comportement sur l'hôte, tel que l'accès aux fichiers, l'exécution des processus et les connexions réseau. Cela couvre des problèmes tels que l'augmentation des privilèges, l'utilisation d'informations d'identification divulguées ou la communication avec des adresses IP ou des domaines malveillants, ainsi que la présence de logiciels malveillants sur vos EC2 instances Amazon et vos charges de travail de conteneur. Pour plus d'informations, consultez la section Surveillance du temps GuardDuty d'exécution dans le guide de GuardDuty l'utilisateur.

Votre administrateur de sécurité active la surveillance du temps d'exécution pour un ou plusieurs comptes dans AWS Organizations for GuardDuty. Ils déterminent également s'il déploie GuardDuty automatiquement l'agent de GuardDuty sécurité lorsque vous utilisez Fargate. Tous vos clusters sont automatiquement protégés et GuardDuty gèrent l'agent de sécurité en votre nom.

Vous pouvez également configurer manuellement l'agent GuardDuty de sécurité dans les cas suivants :

- Vous utilisez des instances de EC2 conteneur
- Vous avez besoin d'un contrôle granulaire pour activer la surveillance du temps d'exécution au niveau du cluster

Pour utiliser la surveillance du temps d'exécution, vous devez configurer les clusters protégés, puis installer et gérer l'agent GuardDuty de sécurité sur vos instances de EC2 conteneur.

Comment fonctionne la surveillance du temps d'exécution avec Amazon ECS

Runtime Monitoring utilise un agent GuardDuty de sécurité léger qui surveille l'activité de la charge de travail Amazon ECS pour déterminer comment les applications demandent, obtiennent l'accès et consomment les ressources système sous-jacentes.

Pour les tâches Fargate, GuardDuty l'agent de sécurité s'exécute comme un conteneur annexe pour chaque tâche.

Pour les instances de EC2 conteneur, l'agent GuardDuty de sécurité s'exécute en tant que processus sur l'instance.

L'agent GuardDuty de sécurité collecte des données à partir des ressources suivantes, puis les envoie GuardDuty à des fins de traitement. Vous pouvez consulter les résultats dans la GuardDuty console. Vous pouvez également les envoyer à un autre fournisseur Services AWS de sécurité ou à un fournisseur de sécurité tiers pour les agréger et les corriger. AWS Security Hub Pour plus d'informations sur la façon de consulter et de gérer les résultats, consultez la section Gérer GuardDuty les résultats d'Amazon dans le guide de GuardDuty l'utilisateur Amazon.

- Réponses des appels d'API Amazon ECS suivants :
  - DescribeClusters

    Les paramètres de réponse incluent la balise Runtime Monitoring (lorsque la balise est définie) lorsque vous utilisez l'`--include TAG`Soption.
  - DescribeTasks

    Pour le type de lancement Fargate, les paramètres de réponse incluent GuardDuty le conteneur du sidecar.

- ListAccountSettings

  Les paramètres de réponse incluent le paramètre du compte Runtime Monitoring, défini par votre administrateur de sécurité.
- Les données d'introspection de l'agent conteneur. Pour de plus amples informations, veuillez consulter Introspection des conteneurs Amazon ECS.
- Point de terminaison des métadonnées de tâche pour le type de lancement :
  - Point de terminaison des métadonnées des tâches Amazon ECS, version 4
  - Point de terminaison de métadonnées de tâches Amazon ECS version 4 pour les tâches sur Fargate

Considérations

Lorsque vous utilisez la surveillance du temps d'exécution, tenez compte des points suivants :

- La surveillance du temps d'exécution a un coût associé. Pour plus d'informations, consultez Amazon GuardDuty Pricing.
- La surveillance du temps d'exécution n'est pas prise en charge sur Amazon ECS Anywhere.
- La surveillance du temps d'exécution n'est pas prise en charge pour le système d'exploitation Windows.
- Lorsque vous utilisez Amazon ECS Exec sur Fargate, vous devez spécifier le nom du conteneur car GuardDuty l'agent de sécurité s'exécute comme un conteneur annexe.
- Vous ne pouvez pas utiliser Amazon ECS Exec sur le conteneur latéral GuardDuty de l'agent de sécurité.
- L'utilisateur IAM qui contrôle la surveillance du temps d'exécution au niveau du cluster doit disposer des autorisations IAM appropriées pour le balisage. Pour plus d'informations, consultez le didacticiel IAM : Définissez les autorisations d'accès aux AWS ressources en fonction des balises du guide de l'utilisateur IAM.
- Les tâches Fargate doivent utiliser un rôle d'exécution de tâches. Ce rôle autorise les tâches à récupérer, mettre à jour et gérer l'agent GuardDuty de sécurité, qui est stocké dans un référentiel privé Amazon ECR, en votre nom.

Utilisation des ressources

La balise que vous ajoutez au cluster est prise en compte dans le quota de balises du cluster.

Le conteneur annexe de l' GuardDuty agent n'est pas pris en compte dans le quota de définition de conteneurs par tâche.

Comme pour la plupart des logiciels de sécurité, il y a une légère surcharge pour GuardDuty. Pour plus d'informations sur les limites de mémoire de Fargate, consultez la section Limites de processeur et de mémoire dans GuardDuty le guide de l'utilisateur. Pour plus d'informations sur les limites de EC2 mémoire d'Amazon, consultez la section Limite de processeur et de mémoire pour GuardDuty l'agent.

Surveillance du temps d'exécution pour les charges de travail Amazon ECS Fargate

Si vous utilisez des instances de EC2 conteneur, vous devez configurer manuellement le Runtime Monitoring. Pour de plus amples informations, veuillez consulter Surveillance de l'exécution des EC2 charges de travail sur Amazon ECS.

Vous pouvez avoir la GuardDuty gestion de l'agent de sécurité sur vos instances de conteneur. Cette option n'est disponible que pour Fargate. Cette option (gestion des GuardDuty agents) est disponible dans GuardDuty

Lorsque vous utilisez la gestion des GuardDuty agents, GuardDuty effectue les opérations suivantes :

- Crée des points de terminaison VPC pour GuardDuty chaque VPC hébergeant un cluster.
- Récupère et installe le dernier agent de GuardDuty sécurité sous forme de conteneur annexe pour toutes les nouvelles tâches Fargate autonomes et les nouveaux déploiements de services.

  Le déploiement d'un nouveau service a lieu la première fois que vous lancez un service ou lorsque vous mettez à jour un service existant avec l'option Forcer un nouveau déploiement.

Activer la surveillance du temps d'exécution pour Amazon ECS

Vous pouvez configurer GuardDuty pour gérer automatiquement l'agent de sécurité pour tous vos clusters Fargate.

Prérequis

Les conditions requises pour utiliser Runtime Monitoring sont les suivantes :

- La version de la plateforme Fargate doit `1.4.0` être ou ultérieure pour Linux.
- Rôles et autorisations IAM pour Amazon ECS :
  - Les tâches Fargate doivent utiliser un rôle d'exécution de tâches. Ce rôle autorise les tâches à récupérer, mettre à jour et gérer l'agent GuardDuty de sécurité en votre nom. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.
  - Vous contrôlez la surveillance du temps d'exécution pour un cluster à l'aide d'une balise prédéfinie. Si vos politiques d'accès limitent l'accès en fonction de balises, vous devez accorder des autorisations explicites à vos utilisateurs IAM pour étiqueter les clusters. Pour plus d'informations, consultez le didacticiel IAM : Définissez les autorisations d'accès aux AWS ressources en fonction des balises du guide de l'utilisateur IAM.
- Connexion au référentiel Amazon ECR :

  L'agent GuardDuty de sécurité est stocké dans un référentiel Amazon ECR. Chaque tâche autonome et de service doit avoir accès au référentiel. Vous pouvez utiliser l'une des options suivantes :
  - Pour les tâches dans les sous-réseaux publics, vous pouvez soit utiliser une adresse IP publique pour la tâche, soit créer un point de terminaison VPC pour Amazon ECR dans le sous-réseau sur lequel la tâche s'exécute. Pour plus d'informations, consultez Points de terminaison d'un VPC d'interface Amazon ECR (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.
  - Pour les tâches dans des sous-réseaux privés, vous pouvez utiliser une passerelle NAT (Network Address Translation) ou créer un point de terminaison VPC pour Amazon ECR dans le sous-réseau sur lequel la tâche s'exécute.

    Pour plus d'informations, consultez Sous-réseau privé et passerelle NAT.
- Vous devez avoir le `AWSServiceRoleForAmazonGuardDuty` rôle pour GuardDuty. Pour plus d'informations, consultez la section Autorisations relatives aux rôles liés à un service GuardDuty dans le guide de GuardDutyl'utilisateur Amazon.
- Tous les fichiers que vous souhaitez protéger avec Runtime Monitoring doivent être accessibles par l'utilisateur root. Si vous avez modifié manuellement les autorisations d'un fichier, vous devez le définir sur`755`.

Les conditions requises pour utiliser la surveillance du temps d'exécution sur les instances de EC2 conteneur sont les suivantes :

- Vous devez utiliser la version `20230929` ou une version ultérieure de l'Amazon ECS-AMI.

- Vous devez exécuter l'agent Amazon ECS selon la version `1.77` ou ultérieure sur les instances de conteneur.
- Vous devez utiliser la version du noyau `5.10` ou une version ultérieure.
- Pour plus d'informations sur les systèmes d'exploitation et les architectures Linux pris en charge, consultez la section Quels sont les modèles d'exploitation et les charges de travail pris en charge par GuardDuty Runtime Monitoring.
- Vous pouvez utiliser Systems Manager pour gérer vos instances de conteneur. Pour plus d'informations, consultez la section Configuration de Systems Manager pour les EC2 instances dans le Guide de AWS Systems Manager Session Manager l'utilisateur.

Procédure

Vous activez la surveillance du temps d'exécution dans GuardDuty. Pour plus d'informations sur l'activation de cette fonctionnalité, consultez la section Enabling Runtime Monitoring dans le guide de GuardDuty l'utilisateur Amazon.

Ajout de la surveillance du temps d'exécution aux tâches Amazon ECS Fargate existantes

Lorsque vous activez la surveillance du temps d'exécution, toutes les nouvelles tâches autonomes et les nouveaux déploiements de services dans le cluster sont automatiquement protégés. Afin de préserver la contrainte d'immuabilité, les tâches existantes ne sont pas affectées.

Prérequis

1. Activez la surveillance du temps d'exécution. Pour de plus amples informations, veuillez consulter Activer la surveillance du temps d'exécution pour Amazon ECS.
2. Les tâches Fargate doivent utiliser un rôle d'exécution de tâches. Ce rôle autorise les tâches à récupérer, mettre à jour et gérer l'agent GuardDuty de sécurité en votre nom. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

Procédure

- Pour protéger immédiatement une tâche, vous devez effectuer l'une des actions suivantes :
  - Pour les tâches autonomes, arrêtez les tâches, puis démarrez-les. Pour plus d'informations, consultez Arrêt d'une tâche Amazon ECS et Exécution d'une application en tant que tâche Amazon ECS

- Pour les tâches faisant partie d'un service, mettez à jour le service avec l'option « forcer un nouveau déploiement ». Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.

Suppression de la surveillance du temps d'exécution d'un cluster Amazon ECS

Vous souhaiterez peut-être exclure certains clusters de la protection, par exemple les clusters que vous utilisez pour les tests. Cela GuardDuty entraîne l'exécution des opérations suivantes sur les ressources du cluster :

- Ne déployez plus l'agent GuardDuty de sécurité sur de nouvelles tâches Fargate autonomes ou sur de nouveaux déploiements de services.

  Afin de préserver la contrainte d'immuabilité, les tâches et les déploiements existants avec la surveillance du temps d'exécution activée ne sont pas affectés.
- Arrêtez la facturation et n'acceptez plus les événements d'exécution des tâches.

Procédure

Procédez comme suit pour supprimer le Runtime Monitoring d'un cluster.

1. Utilisez la console Amazon ECS ou AWS CLI pour définir la clé de `GuardDutyManaged` balise du cluster sur`false`. Pour plus d'informations, consultez la section Mise à jour d'un cluster ou Utilisation des balises à l'aide de la CLI ou de l'API. Utilisez les valeurs suivantes pour le tag.

   > **Note**
   >
   > La clé et la valeur distinguent les majuscules et minuscules et doivent correspondre exactement aux chaînes.

   Clé =`GuardDutyManaged`, Valeur = `false`
2. Supprimez le point de terminaison GuardDuty VPC du cluster. Pour plus d'informations sur la façon de supprimer des points de terminaison VPC, voir Supprimer un point de terminaison d'interface dans le Guide de l'AWS PrivateLink utilisateur.

Supprimer la surveillance du temps d'exécution pour Amazon ECS d'un compte

Lorsque vous ne souhaitez plus utiliser la surveillance du temps d'exécution, désactivez la fonctionnalité dans GuardDuty. Pour plus d'informations sur la façon de désactiver cette fonctionnalité, consultez la section Enabling Runtime Monitoring dans le guide de GuardDuty l'utilisateur Amazon.

GuardDuty effectue les opérations suivantes :

- Supprime les points de terminaison VPC GuardDuty pour chaque VPC hébergeant un cluster.
- Ne déploie plus l'agent de GuardDuty sécurité sur de nouvelles tâches Fargate autonomes ou sur de nouveaux déploiements de services.

  Afin de préserver la contrainte d'immuabilité, les tâches et les déploiements existants ne sont pas affectés tant qu'ils ne sont pas arrêtés, répliqués ou redimensionnés.
- Arrête la facturation et n'accepte plus les événements d'exécution des tâches.

Surveillance de l'exécution des EC2 charges de travail sur Amazon ECS

Utilisez cette option lorsque vous utilisez des EC2 instances pour votre capacité ou lorsque vous avez besoin d'un contrôle granulaire de la surveillance du temps d'exécution au niveau du cluster sur Fargate.

Vous provisionnez les clusters pour la surveillance du temps d'exécution en ajoutant une balise prédéfinie.

Pour les instances de EC2 conteneur, vous téléchargez, installez et gérez l'agent GuardDuty de sécurité.

Pour Fargate GuardDuty , gère l'agent de sécurité en votre nom.

Activer la surveillance du temps d'exécution pour Amazon ECS

Vous pouvez activer la surveillance du temps d'exécution pour les clusters dotés d' EC2 instances ou lorsque vous avez besoin d'un contrôle granulaire de la surveillance du temps d'exécution au niveau du cluster sur Fargate.

Les conditions requises pour utiliser Runtime Monitoring sont les suivantes :

- La version de la plateforme Fargate doit `1.4.0` être ou ultérieure pour Linux.
- Rôles et autorisations IAM pour Amazon ECS :
  - Les tâches Fargate doivent utiliser un rôle d'exécution de tâches. Ce rôle autorise les tâches à récupérer, mettre à jour et gérer l'agent GuardDuty de sécurité en votre nom. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.
  - Vous contrôlez la surveillance du temps d'exécution pour un cluster à l'aide d'une balise prédéfinie. Si vos politiques d'accès limitent l'accès en fonction de balises, vous devez accorder des autorisations explicites à vos utilisateurs IAM pour étiqueter les clusters. Pour plus d'informations, consultez le didacticiel IAM : Définissez les autorisations d'accès aux AWS ressources en fonction des balises du guide de l'utilisateur IAM.
- Connexion au référentiel Amazon ECR :

  L'agent GuardDuty de sécurité est stocké dans un référentiel Amazon ECR. Chaque tâche autonome et de service doit avoir accès au référentiel. Vous pouvez utiliser l'une des options suivantes :
  - Pour les tâches dans les sous-réseaux publics, vous pouvez soit utiliser une adresse IP publique pour la tâche, soit créer un point de terminaison VPC pour Amazon ECR dans le sous-réseau sur lequel la tâche s'exécute. Pour plus d'informations, consultez Points de terminaison d'un VPC d'interface Amazon ECR (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.
  - Pour les tâches dans des sous-réseaux privés, vous pouvez utiliser une passerelle NAT (Network Address Translation) ou créer un point de terminaison VPC pour Amazon ECR dans le sous-réseau sur lequel la tâche s'exécute.

    Pour plus d'informations, consultez Sous-réseau privé et passerelle NAT.
- Vous devez avoir le `AWSServiceRoleForAmazonGuardDuty` rôle pour GuardDuty. Pour plus d'informations, consultez la section Autorisations relatives aux rôles liés à un service GuardDuty dans le guide de GuardDutyl'utilisateur Amazon.
- Tous les fichiers que vous souhaitez protéger avec Runtime Monitoring doivent être accessibles par l'utilisateur root. Si vous avez modifié manuellement les autorisations d'un fichier, vous devez le définir sur`755`.

Les conditions requises pour utiliser la surveillance du temps d'exécution sur les instances de EC2 conteneur sont les suivantes :

- Vous devez utiliser la version `20230929` ou une version ultérieure de l'Amazon ECS-AMI.

- Vous devez exécuter l'agent Amazon ECS selon la version `1.77` ou ultérieure sur les instances de conteneur.
- Vous devez utiliser la version du noyau `5.10` ou une version ultérieure.
- Pour plus d'informations sur les systèmes d'exploitation et les architectures Linux pris en charge, consultez la section Quels sont les modèles d'exploitation et les charges de travail pris en charge par GuardDuty Runtime Monitoring.
- Vous pouvez utiliser Systems Manager pour gérer vos instances de conteneur. Pour plus d'informations, consultez la section Configuration de Systems Manager pour les EC2 instances dans le Guide de AWS Systems Manager Session Manager l'utilisateur.

Vous activez la surveillance du temps d'exécution dans GuardDuty. Pour plus d'informations sur l'activation de cette fonctionnalité, consultez la section Enabling Runtime Monitoring dans le guide de GuardDuty l'utilisateur Amazon.

Ajout de la surveillance du temps d'exécution à un cluster Amazon ECS

Configurez la surveillance du temps d'exécution pour le cluster, puis installez l'agent GuardDuty de sécurité sur vos instances de EC2 conteneur.

Prérequis

1. Activez la surveillance du temps d'exécution. Pour de plus amples informations, veuillez consulter Activer la surveillance du temps d'exécution pour Amazon ECS.
2. Vous contrôlez la surveillance du temps d'exécution pour un cluster à l'aide d'une balise prédéfinie. Si vos politiques d'accès limitent l'accès en fonction de balises, vous devez accorder des autorisations explicites à vos utilisateurs IAM pour étiqueter les clusters. Pour plus d'informations, consultez le didacticiel IAM : Définissez les autorisations d'accès aux AWS ressources en fonction des balises du guide de l'utilisateur IAM.

Procédure

Effectuez les opérations suivantes pour ajouter la surveillance du temps d'exécution à un cluster.

1. Créez un point de terminaison VPC GuardDuty pour chaque VPC de cluster. Pour plus d'informations, consultez la section Création manuelle d'un point de terminaison Amazon VPC dans le guide de l'GuardDuty utilisateur.
2. Configurez les instances de EC2 conteneur.

a. Mettez à jour la version `1.77` ou ultérieure de l'agent Amazon ECS sur les instances de EC2 conteneur du cluster. Pour de plus amples informations, veuillez consulter Mise à jour de l'agent de conteneur Amazon ECS.

b. Installez l'agent GuardDuty de sécurité sur les instances de EC2 conteneur du cluster. Pour plus d'informations, consultez la section Gestion manuelle de l'agent de sécurité sur une EC2 instance Amazon dans le Guide de GuardDuty l'utilisateur.

   Toutes les tâches et tous les déploiements nouveaux et existants sont immédiatement protégés car l'agent GuardDuty de sécurité s'exécute en tant que processus sur l'instance de EC2 conteneur.

3. Utilisez la console Amazon ECS ou AWS CLI pour définir la clé de `GuardDutyManaged` balise du cluster sur`true`. Pour plus d'informations, consultez la section Mise à jour d'un cluster ou Utilisation des balises à l'aide de la CLI ou de l'API. Utilisez les valeurs suivantes pour le tag.

   > **Note**
   >
   > La clé et la valeur distinguent les majuscules et minuscules et doivent correspondre exactement aux chaînes.

   Clé =`GuardDutyManaged`, Valeur = `true`

Ajout de la surveillance du temps d'exécution aux tâches Amazon ECS existantes

Lorsque vous activez la surveillance du temps d'exécution, toutes les nouvelles tâches autonomes et les nouveaux déploiements de services dans le cluster sont automatiquement protégés. Afin de préserver la contrainte d'immuabilité, les tâches existantes ne sont pas affectées.

Prérequis

- Activez la surveillance du temps d'exécution. Pour de plus amples informations, veuillez consulter Activer la surveillance du temps d'exécution pour Amazon ECS.

Procédure

- Pour protéger immédiatement une tâche, vous devez effectuer l'une des actions suivantes :

- Pour les tâches autonomes, arrêtez les tâches, puis démarrez-les. Pour plus d'informations, consultez Arrêt d'une tâche Amazon ECS et Exécution d'une application en tant que tâche Amazon ECS.
- Pour les tâches faisant partie d'un service, mettez à jour le service avec l'option « forcer un nouveau déploiement ». Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.

Suppression de la surveillance du temps d'exécution d'un cluster Amazon ECS

Vous pouvez supprimer la surveillance du temps d'exécution d'un cluster. Cela GuardDuty entraîne l'arrêt de la surveillance de toutes les ressources du cluster.

Pour supprimer la surveillance du temps d'exécution d'un cluster

1. Utilisez la console Amazon ECS ou AWS CLI pour définir la clé de `GuardDutyManaged` balise du cluster sur`false`. Pour plus d'informations, consultez la section Mise à jour d'un cluster ou Utilisation des balises à l'aide de la CLI ou de l'API.

   > **Note**
   >
   > La clé et la valeur distinguent les majuscules et minuscules et doivent correspondre exactement aux chaînes.

   Clé =`GuardDutyManaged`, Valeur = `false`

2. Désinstallez l'agent GuardDuty de sécurité sur vos instances de EC2 conteneur dans le cluster.

   Pour plus d'informations, consultez la section Désinstallation manuelle de l'agent de sécurité dans le guide de l'GuardDuty utilisateur.

3. Supprimez le point de terminaison GuardDuty VPC pour chaque VPC de cluster. Pour plus d'informations sur la façon de supprimer des points de terminaison VPC, voir Supprimer un point de terminaison d'interface dans le Guide de l'AWS PrivateLink utilisateur.

Mettre à jour l'agent GuardDuty de sécurité sur vos instances de conteneur Amazon ECS

Pour plus d'informations sur la mise à jour de l'agent GuardDuty de sécurité sur vos instances de EC2 conteneur, consultez la section Mise à jour GuardDuty de l'agent de sécurité dans le guide de GuardDuty l'utilisateur Amazon.

Supprimer la surveillance du temps d'exécution pour Amazon ECS d'un compte

Lorsque vous ne souhaitez plus utiliser la surveillance du temps d'exécution, désactivez la fonctionnalité dans GuardDuty. Pour plus d'informations sur la façon de désactiver cette fonctionnalité, consultez la section Enabling Runtime Monitoring dans le guide de GuardDuty l'utilisateur Amazon.

Supprimez la surveillance du temps d'exécution de tous les clusters. Pour de plus amples informations, veuillez consulter Suppression de la surveillance du temps d'exécution d'un cluster Amazon ECS.

Résolution des problèmes de surveillance du temps

Vous devrez peut-être résoudre les problèmes ou vérifier que la surveillance du temps d'exécution est activée et exécutée sur vos tâches et vos conteneurs.

Rubriques

- Comment savoir si le Runtime Monitoring est actif sur mon compte ?
- Comment savoir si le Runtime Monitoring est actif sur un cluster ?
- Comment savoir si l'agent de GuardDuty sécurité est exécuté sur une tâche Fargate ?
- Comment savoir si l'agent de GuardDuty sécurité s'exécute sur une instance de EC2 conteneur ?
- Que se passe-t-il lorsqu'il n'existe aucun rôle d'exécution de tâche pour une tâche exécutée sur le cluster ?
- Comment savoir si je dispose des autorisations appropriées pour étiqueter des clusters à des fins de surveillance du temps d'exécution ?
- Que se passe-t-il en cas d'absence de connexion Amazon ECR ?
- Comment puis-je corriger les erreurs liées au manque de mémoire sur mes tâches Fargate après avoir activé la surveillance du temps d'exécution ?

Comment savoir si le Runtime Monitoring est actif sur mon compte ?

Dans la console Amazon ECS, les informations se trouvent sur la page Paramètres du compte.

Vous pouvez également exécuter `list-account-settings` avec l'`effective-settings`option.

```
aws ecs list-account-settings --effective-settings
```

Sortie

Le paramètre dont le nom `guardDutyActivate` et la valeur sont définis sur on indique que le compte est configuré. Vous devez vérifier auprès de votre GuardDuty administrateur si la gestion est automatique ou manuelle.

```
{
    "setting": {
        "name": "guardDutyActivate",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type": "aws-managed"
    }
}
```

Comment savoir si le Runtime Monitoring est actif sur un cluster ?

Vous pouvez consulter les statistiques de couverture dans la GuardDuty console. Cela inclut les informations relatives aux ressources Amazon ECS associées à votre propre compte ou à vos comptes de membres, c'est-à-dire le pourcentage de clusters sains par rapport à tous les clusters sélectionnés Région AWS. Cela inclut la couverture des clusters qui utilisent les types Fargate et Launch. EC2 Pour plus d'informations, consultez la section Révision des statistiques de couverture dans le guide de GuardDuty l'utilisateur Amazon.

Comment savoir si l'agent de GuardDuty sécurité est exécuté sur une tâche Fargate ?

L'agent GuardDuty de sécurité fonctionne comme un conteneur annexe pour les tâches Fargate.

Dans la console Amazon ECS, le sidecar s'affiche sous Conteneurs sur la page des détails de la tâche.

Vous pouvez exécuter `describe-tasks` et rechercher le conteneur dont le nom est défini sur `aws-gd-agent` et le LastStatus défini sur. `RUNNING`

L'exemple suivant montre la sortie du cluster par défaut pour la tâche`aws:ecs:us-east-1:123456789012:task/0b69d5c0-d655-4695-98cd-5d2d5EXAMPLE`.

```
aws ecs describe-tasks --cluster default --tasks aws:ecs:us-
east-1:123456789012:task/0b69d5c0-d655-4695-98cd-5d2d5EXAMPLE
```

Sortie

Le conteneur nommé `gd-agent` est dans l'`RUNNING`état.

```
"containers": [
      {
        "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/4df26bb4-
f057-467b-a079-96167EXAMPLE",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/0b69d5c0-
d655-4695-98cd-5d2d5EXAMPLE",
        "lastStatus": "RUNNING",
        "healthStatus": "UNKNOWN",
        "memory": "string",
        "name": "aws-gd-agent"
      }
    ]
```

Comment savoir si l'agent de GuardDuty sécurité s'exécute sur une instance de EC2 conteneur ?

Exécutez la commande suivante pour afficher l'état :

```
sudo systemctl status amazon-guardduty-agent
```

Le fichier journal se trouve à l'emplacement suivant :

```
/var/log/amzn-guardduty-agent
```

Que se passe-t-il lorsqu'il n'existe aucun rôle d'exécution de tâche pour une tâche exécutée sur le cluster ?

Pour les tâches Fargate, la tâche démarre sans le conteneur annexe de GuardDuty l'agent de sécurité. Le GuardDuty tableau de bord indiquera que la tâche n'est pas protégée dans le tableau de bord des statistiques de couverture.

Comment savoir si je dispose des autorisations appropriées pour étiqueter des clusters à des fins de surveillance du temps d'exécution ?

Pour étiqueter un cluster, vous devez avoir l'`ecs:TagResource`action à la fois pour `CreateCluster` et`UpdateCluster`.

Voici un extrait d'un exemple de politique.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ecs:TagResource"
      ],
      "Resource": "*",
      "Condition": {
         "StringEquals": {
             "ecs:CreateAction" : "CreateCluster",
             "ecs:CreateAction" : "UpdateCluster",
          }
       }
    }
  ]
}
```

Que se passe-t-il en cas d'absence de connexion Amazon ECR ?

Pour les tâches Fargate, la tâche démarre sans le conteneur annexe de GuardDuty l'agent de sécurité. Le GuardDuty tableau de bord indiquera que la tâche n'est pas protégée dans le tableau de bord des statistiques de couverture.

Comment puis-je corriger les erreurs liées au manque de mémoire sur mes tâches Fargate après avoir activé la surveillance du temps d'exécution ?

L'agent GuardDuty de sécurité est un processus léger. Cependant, le processus consomme toujours des ressources en fonction de l'importance de la charge de travail. Nous vous recommandons d'utiliser des outils de suivi des ressources des conteneurs, tels que Amazon CloudWatch Container Insights, pour organiser GuardDuty les déploiements dans votre cluster. Ces outils vous aident à découvrir le profil de consommation de l'agent de GuardDuty sécurité pour vos applications. Vous

pouvez ensuite ajuster la taille de votre tâche Fargate, si nécessaire, pour éviter tout risque de manque de mémoire.

Surveillez les conteneurs Amazon ECS avec ECS Exec

Avec Amazon ECS Exec, vous pouvez interagir directement avec les conteneurs sans avoir besoin d'interagir avec le système d'exploitation du conteneur hôte, d'ouvrir des ports entrants ou de gérer les clés SSH au préalable. Vous pouvez utiliser ECS Exec pour exécuter des commandes ou transférer un shell vers un conteneur exécuté sur une EC2 instance Amazon ou sur AWS Fargate. Cela facilite la collecte des informations de diagnostic et la résolution rapide des erreurs. Par exemple, dans le cadre d'un développement, vous pouvez utiliser ECS Exec pour interagir facilement avec différents processus dans vos conteneurs et dépanner vos applications. Et dans les scénarios de production, vous pouvez l'utiliser pour accéder facilement à vos conteneurs afin de résoudre les problèmes.

Vous pouvez exécuter des commandes dans un conteneur Linux ou Windows en cours d'exécution à l'aide d'ECS Exec à partir de l'API Amazon ECS AWS Command Line Interface (AWS CLI) ou de la AWS SDKs CLI AWS Copilot. Pour plus de détails sur l'utilisation d'ECS Exec, ainsi qu'une présentation vidéo sur l'utilisation de la CLI AWS Copilot, consultez la documentation Copilot. GitHub

Vous pouvez également utiliser ECS Exec pour maintenir des politiques de contrôle d'accès plus strictes. En activant cette fonction de manière sélective, vous pouvez contrôler qui peut exécuter des commandes et sur quelles tâches. Avec un journal de chaque commande et de ses résultats, vous pouvez utiliser ECS Exec pour voir quelles tâches ont été exécutées et CloudTrail pour vérifier qui a accédé à un conteneur.

Considérations

Pour cette rubrique, vous devez connaître les aspects suivants liés à l'utilisation d'ECS Exec :

- ECS Exec n'est actuellement pas pris en charge avec le AWS Management Console.

  La page de détails des clusters affiche la clé de journalisation et la clé gérée par le AWS KMS client utilisées pour chiffrer les données entre le client local et le conteneur.
- ECS Exec risque de ne pas fonctionner comme prévu s'il est exécuté sur des systèmes d'exploitation non pris en charge par Systems Manager. Pour plus d'informations sur les systèmes d'exploitation pris en charge, reportez-vous à la section Types de systèmes d'exploitation du Guide de AWS Systems Manager l'utilisateur.

- ECS Exec est pris en charge pour les tâches exécutées sur l'infrastructure suivante :
  - Conteneurs Linux sur Amazon EC2 sur n'importe quelle AMI optimisée pour Amazon ECS, y compris Bottlerocket
  - Conteneurs Linux et Windows sur des instances externes (Amazon ECS Anywhere)
  - Conteneurs Linux et Windows sur AWS Fargate
  - Conteneurs Windows sur Amazon EC2 sur les modèles Windows suivants optimisés pour Amazon ECS AMIs (avec la version de l'agent de conteneur `1.56` ou ultérieure) :
    - AMI de Windows Server 2022 Full optimisée pour Amazon ECS
    - AMI de Windows Server 2022 Core optimisée pour Amazon ECS
    - AMI de Windows Server 2019 Full optimisée pour Amazon ECS
    - AMI de Windows Server 2019 Core optimisée pour Amazon ECS
    - AMI de Windows Server 20H2 Core optimisée pour Amazon ECS
- Si vous avez configuré un proxy HTTP pour votre tâche, définissez la variable d'`NO_PROXY`environnement sur `"NO_PROXY=169.254.169.254,169.254.170.2"` afin de contourner le proxy pour les métadonnées d' EC2 instance et le trafic des rôles IAM. Si vous ne configurez pas la variable d'`NO_PROXY`environnement, des échecs peuvent survenir lors de la récupération des métadonnées de l'instance ou des informations d'identification du rôle IAM à partir du point de terminaison des métadonnées du conteneur. La définition de la variable d'`NO_PROXY`environnement comme recommandé filtre les métadonnées et le trafic IAM afin que les demandes `169.254.169.254 and 169.254.170.2` ne passent pas par le `HTTP` proxy.
- ECS Exec et Amazon VPC
  - Si vous utilisez l'interface des points de terminaison Amazon VPC avec Amazon ECS, vous devez créer l'interface des points de terminaison Amazon VPC pour le Gestionnaire de sessions Systems Manager (`ssmmessages`). Pour plus d'informations sur les points de terminaison VPC de Systems Manager, consultez la section Utiliser AWS PrivateLink pour configurer un point de terminaison VPC pour Session Manager dans le guide de l'utilisateur.AWS Systems Manager
  - Si vous utilisez les points de terminaison Amazon VPC d'interface avec Amazon ECS, et que vous utilisez AWS KMS key pour le chiffrement, alors vous devez créer le point de terminaison Amazon VPC d'interface pour AWS KMS key. Pour plus d'informations, consultez Configuration à AWS KMS key via un point de terminaison de VPC dans le Guide du développeur AWS Key Management Service .
  - Lorsque vous avez des tâches exécutées sur des EC2 instances Amazon, utilisez le mode `awsvpc` réseau. Si vous n'avez pas accès à Internet (par exemple si vous n'êtes pas configuré pour utiliser une passerelle NAT), vous devez créer l'interface Amazon VPC endpoints pour le

gestionnaire de session Systems Manager (). `ssmmessages` Pour plus d'informations sur les considérations relatives au mode réseau `awsvpc`, veuillez consulter Considérations (langue française non garantie). Pour plus d'informations sur les points de terminaison VPC de Systems Manager, consultez la section Utiliser AWS PrivateLink pour configurer un point de terminaison VPC pour Session Manager dans le guide de l'utilisateur.AWS Systems Manager

- ECS Exec et SSM
  - Lorsqu'un utilisateur exécute des commandes sur un conteneur à l'aide d'ECS Exec, ces commandes sont exécutées en tant qu'utilisateur `root`. Le SSM Agent et ses processus enfants s'exécutent en tant qu'utilisateur racine même lorsque vous spécifiez un ID d'utilisateur pour le conteneur.
  - L'agent SSM nécessite que le système de fichiers conteneur puisse être écrit dans le système de fichiers pour créer les répertoires et les fichiers requis. Par conséquent, il n'est pas possible de rendre le système de fichiers racine en lecture seule à l'aide du paramètre de définition de tâche `readonlyRootFilesystem` ou de toute autre méthode.
  - Il est possible de démarrer des sessions SSM en dehors de l'action `execute-command`, mais les sessions ne sont pas journalisées et sont comptabilisées afin de vérifier qu'elles ne dépassent pas la limite de session. Nous vous recommandons de limiter cet accès en refusant l'action `ssm:start-session` à l'aide d'une stratégie IAM. Pour de plus amples informations, veuillez consulter Limitation de l'accès à l'action StartSession.
- Les fonctionnalités suivantes s'exécutent en tant que conteneur de sidecar. Par conséquent, vous devez spécifier le nom du conteneur sur lequel exécuter la commande.
  - Surveillance d'exécution
  - Service Connect
- Les utilisateurs peuvent exécuter toutes les commandes qui sont disponibles dans le contexte de conteneur. Les actions suivantes peuvent entraîner des processus orphelins et zombies : arrêt du processus principal du conteneur, arrêt de l'agent de commande et suppression des dépendances. Pour nettoyer les processus zombie, nous vous recommandons d'ajouter l'indicateur `initProcessEnabled` à votre définition de tâche.
- ECS Exec utilise une partie du processeur et de la mémoire. Vous devez en tenir compte lorsque vous spécifiez les allocations de ressources CPU et mémoire dans votre définition de tâche.
- Vous devez utiliser la AWS CLI version `1.22.3` ou une version ultérieure ou une AWS CLI version `2.3.6` ou une version ultérieure. Pour plus d'informations sur la mise à jour du AWS CLI, voir Installation ou mise à jour de la AWS CLI dernière version du Guide de l'AWS Command Line Interface utilisateur, version 2.

- Vous ne pouvez avoir qu'une seule session ECS Exec par espace de noms d'ID de processus (PID). Si vous partagez un espace de noms PID dans une tâche, vous ne pouvez démarrer des sessions ECS Exec que dans un seul conteneur.
- La session ECS Exec a une valeur de délai d'inactivité de 20 minutes. Cette valeur ne peut pas être modifiée.
- Vous ne pouvez pas activer ECS Exec pour les tâches existantes. Vous ne pouvez l'activer que pour les nouvelles tâches.
- Vous ne pouvez pas utiliser ECS Exec lorsque vous lancez une tâche sur un cluster qui utilise un dimensionnement géré avec un placement asynchrone (lancement d'une tâche sans instance). `run-task`
- Vous ne pouvez pas exécuter ECS Exec sur des conteneurs Microsoft Nano Server.

Prérequis

Avant de commencer à utiliser ECS Exec, assurez-vous d'avoir effectué les actions suivantes :

- Installation et configuration de l' AWS CLI. Pour plus d'informations, voir Commencer avec le AWS CLI.
- Installez le plugin Session Manager pour AWS CLI. Pour de plus amples informations, veuillez consulter Install the Session Manager plugin for the AWS CLI.
- Vous devez utiliser un rôle de tâche doté des autorisations appropriées pour ECS Exec. Pour plus d'informations, veuillez consulter Rôle IAM de tâche (langue française non garantie).
- ECS Exec a des exigences de version selon que vos tâches sont hébergées sur Amazon EC2 ou AWS Fargate :
  - Si vous utilisez Amazon EC2, vous devez utiliser une AMI optimisée pour Amazon ECS publiée après le 20 janvier 2021, avec une version d'agent 1.50.2 ou supérieure. Pour plus d'informations, consultez Amazon ECS optimisé AMIs.
  - Si vous utilisez AWS Fargate, vous devez utiliser une version de plate-forme `1.4.0` ou supérieure (Linux) ou `1.0.0` (Windows). Pour plus d'informations, consultez Versions de plateforme AWS Fargate.

Architecture

ECS Exec utilise le gestionnaire de session AWS Systems Manager (SSM) pour établir une connexion avec le conteneur en cours d'exécution et utilise des politiques AWS Identity and Access

Management (IAM) pour contrôler l'accès aux commandes en cours d'exécution dans un conteneur en cours d'exécution. Ceci est rendu possible par le montage lié des fichiers binaires du SSM Agent nécessaires dans le conteneur. L'Amazon ECS ou l' AWS Fargate agent est chargé de démarrer l'agent principal SSM dans le conteneur en même temps que le code de votre application. Pour plus d'informations, consultez Systems Manager Session Manager.

Vous pouvez vérifier quel utilisateur a accédé au conteneur à l'aide de l'`ExecuteCommand`événement AWS CloudTrail et enregistrer chaque commande (et sa sortie) dans Amazon S3 ou Amazon CloudWatch Logs. Pour chiffrer les données entre le client local et le conteneur avec votre propre clé de chiffrement, vous devez fournir la clé AWS Key Management Service (AWS KMS).

Utilisation d'ECS Exec

Modifications facultatives de la définition de tâche

Si vous définissez le paramètre de définition de tâche `initProcessEnabled` sur`true`, le processus d'initialisation démarre dans le conteneur. Cela supprime tous les processus enfants de l'agent SSM zombie trouvés. Voici un exemple.

```
{
    "taskRoleArn": "ecsTaskRole",
    "networkMode": "awsvpc",
    "requiresCompatibilities": [
        "EC2",
        "FARGATE"
    ],
    "executionRoleArn": "ecsTaskExecutionRole",
    "memory": ".5 gb",
    "cpu": ".25 vcpu",
    "containerDefinitions": [
        {
            "name": "amazon-linux",
            "image": "amazonlinux:latest",
            "essential": true,
            "command": ["sleep","3600"],
            "linuxParameters": {
                "initProcessEnabled": true
            }
        }
```

```
    ],
    "family": "ecs-exec-task"
}
```

Activation d'ECS Exec pour vos tâches et services

Vous pouvez activer la fonctionnalité ECS Exec pour vos services et tâches autonomes en spécifiant l'`--enable-execute-command`indicateur lorsque vous utilisez l'une des AWS CLI commandes suivantes : `create-service`, `update-servicestart-task`, ou. `run-task`

Par exemple, si vous exécutez la commande suivante, la fonctionnalité ECS Exec est activée pour un service nouvellement créé qui s'exécute sur Fargate. Pour plus d'informations sur la création de services, consultez create-service.

```
aws ecs create-service \
    --cluster cluster-name \
    --task-definition task-definition-name \
    --enable-execute-command \
    --service-name service-name \
    --launch-type FARGATE \
     --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
 \
    --desired-count 1
```

Après avoir activé ECS Exec pour une tâche, vous pouvez exécuter la commande suivante pour confirmer que la tâche est prête à être utilisée. Si la propriété `lastStatus` de `ExecuteCommandAgent` est répertoriée comme `RUNNING` et que la propriété `enableExecuteCommand` est définie sur `true`, alors votre tâche est prête.

```
aws ecs describe-tasks \
    --cluster cluster-name \
    --tasks task-id
```

L'extrait de sortie suivant est un exemple de ce que vous pouvez voir.

```
{
    "tasks": [
        {
            ...
```

```
            "containers": [
                {
                    ...
                    "managedAgents": [
                        {
                            "lastStartedAt": "2021-03-01T14:49:44.574000-06:00",
                            "name": "ExecuteCommandAgent",
                            "lastStatus": "RUNNING"
                        }
                    ]
                }
            ],
            ...
            "enableExecuteCommand": true,
            ...
        }
    ]
}
```

Exécution de commandes avec ECS Exec

Une fois que vous avez confirmé que `ExecuteCommandAgent` est en cours d'exécution, vous pouvez ouvrir un shell interactif sur votre conteneur à l'aide de la commande suivante. Si votre tâche contient plusieurs conteneurs, vous devez spécifier le nom du conteneur à l'aide de l'indicateur `--container`. Amazon ECS ne prend en charge que le lancement de sessions interactives. Vous devez donc utiliser l'indicateur `--interactive`.

La commande suivante exécutera une `/bin/sh` commande interactive sur un conteneur nommé *`container-name`* d'après une tâche avec un ID de*`task-id`*.

*`task-id`*Il s'agit du nom de ressource Amazon (ARN) de la tâche.

```
aws ecs execute-command --cluster cluster-name \
    --task task-id \
    --container container-name \
    --interactive \
    --command "/bin/sh"
```

Journalisation à l'aide d'ECS Exec

Activer la connexion à vos tâches et services

> **Important**
>
> Pour plus d'informations sur la CloudWatch tarification, consultez la section CloudWatch Tarification. Amazon ECS propose également des métriques de surveillance sans coûts supplémentaires. Pour de plus amples informations, veuillez consulter Surveillez Amazon ECS à l'aide de CloudWatch.

Amazon ECS fournit une configuration par défaut pour les commandes de journalisation exécutées à l'aide d'ECS Exec. Par défaut, les journaux sont envoyés à CloudWatch Logs à l'aide du pilote de `awslogs` journal configuré dans la définition de votre tâche. Si vous souhaitez fournir une configuration personnalisée, le AWS CLI support inclut un `--configuration` indicateur pour les `update-cluster` commandes `create-cluster` et. L'image du conteneur nécessite `script` et `cat` doit être installée afin que les journaux de commandes soient correctement chargés sur Amazon S3 ou CloudWatch Logs. Pour de plus amples informations sur la création de clusters, consultez create-cluster.

> **Note**
>
> Cette configuration ne gère que la journalisation de la session `execute-command`. Cela n'affecte pas la journalisation de votre application.

L'exemple suivant crée un cluster, puis enregistre la sortie dans vos CloudWatch journaux LogGroup nommés `cloudwatch-log-group-name` et dans votre compartiment Amazon S3 nommés`s3-bucket-name`.

Vous devez utiliser une clé gérée par le AWS KMS client pour chiffrer le groupe de journaux lorsque vous définissez l'`CloudWatchEncryptionEnabled`option sur. `true` Pour plus d'informations sur le chiffrement du groupe de journaux, voir Chiffrer les données des CloudWatch journaux dans Logs using AWS Key Management Service, dans le Guide de l'Amazon CloudWatch Logs utilisateur.

```
aws ecs create-cluster \
```

```
    --cluster-name cluster-name \
    --configuration executeCommandConfiguration="{ \
        kmsKeyId=string, \
        logging=OVERRIDE, \
        logConfiguration={ \
            cloudWatchLogGroupName=cloudwatch-log-group-name, \
            cloudWatchEncryptionEnabled=true, \
            s3BucketName=s3-bucket-name, \
            s3EncryptionEnabled=true, \
            s3KeyPrefix=demo \
        } \
    }"
```

La propriété `logging` détermine le comportement de la capacité de journalisation d'ECS Exec :

- `NONE`: la journalisation est désactivée.
- `DEFAULT`: les journaux sont envoyés au `awslogs` pilote configuré. Si le pilote n'est pas configuré, aucun journal n'est enregistré.
- `OVERRIDE`: les journaux sont envoyés au compartiment Amazon CloudWatch Logs fourni LogGroup, au compartiment Amazon S3, ou aux deux.

Autorisations IAM requises pour Amazon CloudWatch Logs ou Amazon S3 Logging

Pour activer la journalisation, le rôle de tâche Amazon ECS référencé dans votre définition de tâche doit disposer d'autorisations supplémentaires. Ces autorisations supplémentaires peuvent être ajoutées en tant que stratégie au rôle de tâche. Ils sont différents selon que vous dirigez vos journaux vers Amazon CloudWatch Logs ou Amazon S3.

Amazon CloudWatch Logs

L'exemple de politique suivant ajoute les autorisations Amazon CloudWatch Logs requises.

JSON

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
               "logs:CreateLogStream",
               "logs:DescribeLogStreams",
               "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:arn:aws:: 111122223333:log-group:/aws/
ecs/cloudwatch-log-group-name:*"
        }
   ]
}
```

Amazon S3

L'exemple suivant de stratégie ajoute les autorisations Amazon S3 requises.

JSON

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "s3:GetBucketLocation"
            ],
            "Resource": "*"
        },
        {
           "Effect": "Allow",
           "Action": [
               "s3:GetEncryptionConfiguration"
           ],
           "Resource": "arn:aws:s3:::s3-bucket-name"
        },
        {
```

```
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::s3-bucket-name/*"
        }
    ]
  }
```

Autorisations IAM requises pour le chiffrement à l'aide des vôtres AWS KMS key (clé KMS)

Par défaut, les données transférées entre votre client local et le conteneur utilisent le cryptage TLS 1.2 qui AWS fournit. Pour chiffrer davantage les données à l'aide de votre propre clé KMS, vous devez créer une clé KMS et ajouter l'autorisation `kms:Decrypt` à votre rôle IAM de tâche. Votre conteneur utilise cette autorisation pour déchiffrer les données. Pour plus d'informations sur la création d'une clé KMS, consultez Création de clés.

Vous ajoutez la politique intégrée suivante à votre rôle IAM de tâche qui nécessite les AWS KMS autorisations. Pour de plus amples informations, veuillez consulter Autorisations ECS Exec.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource":
 "arn:aws:kms:arn:aws::111122223333:key/12345678-1234-1234-1234-123456789012"
        }
    ]
}
```

Pour que les données soient chiffrées à l'aide de votre propre clé KMS, l'utilisateur ou le groupe utilisant l'action `execute-command` doit être accordée à l'autorisation `kms:GenerateDataKey`.

L'exemple de stratégie suivant pour votre utilisateur ou groupe contient l'autorisation requise pour utiliser votre propre clé KMS. Vous devez spécifier l'Amazon Resource Name (ARN) de votre clé KMS.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:arn:aws::111122223333:key/key"
        }
    ]
}
```

Utilisation de stratégies IAM pour limiter l'accès à ECS Exec

Vous limitez l'accès des utilisateurs à l'action de l'API d'exécution de commande en utilisant une ou plusieurs des clés de condition de politique IAM suivantes :

- `aws:ResourceTag/clusterTagKey`
- `ecs:ResourceTag/clusterTagKey`
- `aws:ResourceTag/taskTagKey`
- `ecs:ResourceTag/taskTagKey`
- `ecs:container-name`
- `ecs:cluster`
- `ecs:task`
- `ecs:enable-execute-command`

Avec l'exemple de stratégie IAM suivant, les utilisateurs peuvent exécuter des commandes dans des conteneurs qui s'exécutent dans des tâches avec une balise qui possède une clé `environment` et une valeur `development`, dans un cluster nommé `cluster-name`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ExecuteCommand",
                "ecs:DescribeTasks"
            ],
            "Resource": [
                   "arn:aws:ecs:arn:aws::111122223333:task/cluster-name/*",
                   "arn:aws:ecs:arn:aws::111122223333:cluster/cluster-name"
            ],
            "Condition": {
                "StringEquals": {
                    "ecs:ResourceTag/environment": "development"
                }
            }
        }
    ]
}
```

Avec l'exemple de politique IAM suivant, les utilisateurs ne peuvent pas utiliser l'API `execute-command` lorsque le nom du conteneur est `production-app`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ecs:ExecuteCommand"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
```

```
                    "ecs:container-name": "production-app"
                }
            }
        }
    ]
}
```

Avec la stratégie IAM suivante, les utilisateurs ne peuvent lancer des tâches que lorsque ECS Exec est désactivé.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:CreateService",
                "ecs:UpdateService"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:enable-execute-command": "false"
                }
            }
        }
    ]
}
```

> **Note**
>
> Étant donné que l'action d'API `execute-command` contient uniquement des ressources de tâche et de cluster dans une reqûete, seules les balises de cluster et de tâche sont évaluées.

Pour plus d'informations sur les clés de condition de stratégie IAM, consultez Actions, ressources et clés de condition pour Amazon Elastic Container Service dans la Référence de l'autorisation de service.

Limitation de l'accès à l'action StartSession

Bien qu'il soit possible de démarrer des sessions SSM sur votre conteneur en dehors d'ECS Exec, cela pourrait entraîner la non-journalisation des sessions. Les sessions démarrées en dehors d'ECS Exec sont également comptabilisées dans le quota de session. Nous vous recommandons de limiter cet accès en refusant l'action `ssm:start-session` directement pour vos tâches Amazon ECS à l'aide d'une stratégie IAM. Vous pouvez refuser l'accès à toutes les tâches Amazon ECS ou à des tâches spécifiques en fonction des balises utilisées.

Voici un exemple de stratégie IAM qui refuse l'accès à l'action `ssm:start-session` pour les tâches dans toutes les régions avec un nom de cluster spécifié. Si vous le souhaitez, vous pouvez inclure un caractère générique avec l'option *`cluster-name`*.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": [
            "arn:aws:ecs:arn:aws:;: 111122223333:task/cluster-name/*"
            ]
        }
    ]
}
```

Voici un exemple de stratégie IAM qui refuse l'accès à l'action `ssm:start-session` sur les ressources dans toutes les régions étiquetées avec la clé de balise `Task-Tag-Key` et la valeur de balise`Exec-Task`.

JSON

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": "arn:aws:ecs:*:*:task/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Task-Tag-Key": "Exec-Task"
                }
            }
        }
    ]
}
```

Pour obtenir de l'aide concernant les problèmes que vous pourriez rencontrer lors de l'utilisation d'Amazon ECS Exec, consultez la section Résolution des problèmes liés à Exec.

AWS Compute Optimizer recommandations pour Amazon ECS

AWS Compute Optimizer génère des recommandations relatives à la taille des tâches et des conteneurs Amazon ECS. Pour plus d'informations, consultez Présentation d' AWS Compute Optimizer dans le Guide de l'utilisateur AWS Compute Optimizer .

Recommandations relatives à la taille des tâches et des conteneurs pour les services Amazon ECS sur AWS Fargate

AWS Compute Optimizer génère des recommandations pour les services Amazon ECS sur AWS Fargate. AWS Compute Optimizer recommande la taille du processeur et de la mémoire des tâches ainsi que les tailles du processeur du conteneur, de la mémoire du conteneur et de la mémoire du conteneur. Ces recommandations sont affichées sur les pages suivantes de la console Compute Optimizer.

- Recommandations pour les services Amazon ECS sur la page Fargate
- Services Amazon ECS sur la page de détails de Fargate

Pour plus d'informations, consultez Affichage des recommandations relatives aux services Amazon ECS sur Fargate dans le Guide de l'utilisateur AWS Compute Optimizer .

Dépannage d'Amazon ECS

Vous devrez peut-être résoudre des problèmes liés à vos équilibreurs de charge, à vos tâches, à vos services ou à vos instances de conteneur. Ce chapitre vous aide à trouver les informations de diagnostic à partir de l'agent de conteneur Amazon ECS, du démon Docker sur l'instance de conteneur et du journal des événements de service de la console Amazon ECS.

Pour plus d'informations sur les tâches arrêtées, consultez l'une des rubriques suivantes.

Action	En savoir plus
Résolvez les erreurs liées aux tâches arrêtées.	L'affichage d'Amazon ECS a permis de stopper les erreurs de tâches
Afficher les erreurs liées aux tâches arrêtées.	Résoudre les erreurs liées aux tâches arrêtées par Amazon ECS
Vérifiez les codes d'erreur des tâches arrêtées.	Messages d'erreur relatifs aux tâches interrompues par Amazon ECS
Vérifiez les erreurs liées aux CannotPullContainer tâches.	CannotPullContainer erreurs de tâche dans Amazon ECS
Afficher les demandes de rôle IAM pour les tâches.	Afficher les demandes de rôle IAM pour les tâches Amazon ECS

Pour plus d'informations sur les erreurs de service, consultez l'un des documents suivants.

Action	En savoir plus
Afficher les messages relatifs aux événements de service.	Affichage des messages relatifs aux événements du service Amazon ECS

Action	En savoir plus	
Passez en revue les messages relatifs aux événements de service.	Messages relatifs aux événements du service Amazon ECS	
Passez en revue les problèmes liés à l'équilibreur de charge	Résolution des problèmes liés aux équilibreurs de charge de service dans Amazon ECS	
Passez en revue les problèmes de mise à l'échelle automatique du service.	Résolution des problèmes liés au dimensionnement automatique du service dans Amazon ECS	

Pour plus d'informations sur les erreurs de définition des tâches, reportez-vous à l'une des sections suivantes.

Action	En savoir plus	
Résolvez l'erreur de mémoire de définition de tâche.	Résoudre les erreurs de processeur ou de mémoire non valides liées à la définition des tâches Amazon ECS	

Pour plus d'informations sur les erreurs des agents Amazon ECS, consultez l'un des documents suivants.

Action	En savoir plus	
Consultez les journaux des agents de conteneurs Amazon ECS.	Afficher les journaux des agents de conteneurs Amazon ECS	

Action	En savoir plus
Découvrez comment collecter les journaux Amazon ECS.	Collecte des journaux de conteneurs avec le collecteur de journaux Amazon ECS
Récupérez les informations de diagnostic avec l'agent Amazon ECS.	Récupérez les informations de diagnostic d'Amazon ECS grâce à l'introspection de l'agent

Pour plus d'informations sur les erreurs Docker, consultez l'un des documents suivants.

Action	En savoir plus
Utilisez les diagnostics Docker.	Diagnostic Docker dans Amazon ECS
Activez le mode de débogage Docker.	Configuration de la sortie détaillée du démon Docker dans Amazon ECS
Résolvez l'erreur 500 de l'API Docker.	Résoudre les problèmes liés au Docker dans `API error (500): devmapper` Amazon ECS

Pour plus d'informations sur les erreurs d'ECS Exec et d'Amazon ECS Anywhere, consultez l'un des documents suivants.

Action	En savoir plus
Résoudre les problèmes d'ECS Exec.	Résoudre les problèmes liés à Amazon ECS Exec

Action	En savoir plus
Résolvez les problèmes liés à Amazon ECS Anywhere.	Résoudre les problèmes liés à Amazon ECS Anywhere

Pour plus d'informations sur les problèmes de limitation, consultez l'un des documents suivants.

Action	En savoir plus
En savoir plus sur les quotas de régulation de Fargate.	AWS Fargate limitation des quotas
Découvrez les meilleure s pratiques relatives à la régulation d'Amazon ECS.	Gérer les problèmes de régulation d'Amazon ECS

Pour plus d'informations sur les erreurs d'API, consultez l'un des documents suivants.

Action	En savoir plus
Résolvez les erreurs d'API.	Raisons de l'échec de l'API Amazon ECS

Résoudre les erreurs liées aux tâches arrêtées par Amazon ECS

Lorsque votre tâche ne démarre pas, un message d'erreur s'affiche dans la console et dans les paramètres `describe-tasks` de sortie (`stoppedReasonetstoppedCode`).

Vous pouvez consulter les tâches arrêtées dans la console pendant une heure. Pour voir les tâches arrêtées, vous devez modifier l'option de filtre. Pour de plus amples informations, veuillez consulter L'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

Les pages suivantes fournissent des informations sur les tâches arrêtées.

- Découvrez les modifications apportées aux messages d'erreur relatifs aux tâches arrêtées.

  Amazon ECS a arrêté la mise à jour des messages d'erreur des tâches

- Consultez les tâches que vous avez arrêtées afin d'obtenir des informations sur leur cause.

  L'affichage d'Amazon ECS a permis de stopper les erreurs de tâches
- Découvrez les messages d'erreur relatifs aux tâches arrêtées et les causes possibles de ces erreurs.

  Messages d'erreur relatifs aux tâches interrompues par Amazon ECS
- Découvrez comment vérifier la connectivité des tâches arrêtées et corriger les erreurs.

  Vérification de l'arrêt de la connectivité des tâches par Amazon ECS

Amazon ECS a arrêté la mise à jour des messages d'erreur des tâches

À compter du 14 juin 2024, l'équipe Amazon ECS modifiera les messages d'erreur relatifs aux tâches arrêtées, comme décrit dans les tableaux suivants. Ils ne `stopCode` changeront pas. Si vos applications dépendent de chaînes de messages d'erreur exactes, vous devez les mettre à jour avec les nouvelles chaînes. Pour obtenir de l'aide en cas de questions ou de problèmes, contactez AWS Support.

> **Note**
>
> Nous vous recommandons de ne pas vous fier aux messages d'erreur pour votre automatisation, car ils sont susceptibles de changer.

CannotPullContainerError

Ancien message d'erreur	Nouveau messages d'erreur	
CannotPullContainerError: Réponse d'erreur du démon : accès au pull refusé pour*repository* , le référentiel n'existe pas ou peut nécessiter une « connexion docker » : refusé : utilisateur : *roleARN*	• CannotPullContainerError: La tâche ne peut pas extraire l'image. Vérifiez que le rôle dispose des autorisations nécessaires pour extraire des images du registre. Réponse d'erreur du démon : accès au pull	

Ancien message d'erreur	Nouveau messages d'erreur	
	refusé pour*repository* , le référentiel n'existe pas ou peut nécessiter une « connexion docker » : refusé : l'utilisateur : n'*roleARN*est pas autorisé à effectuer : ecr : sur la ressource : *image* car aucune politique basée BatchGetImage sur l'identité n'autorise l'action ecr :. BatchGetImage • CannotPullContainerError: La tâche ne peut pas extraire l'image. Vérifiez si l'image existe. Réponse d'erreur du démon : accès au pull refusé pour*repository* , le référentiel n'existe pas ou peut nécessiter une « connexion docker » : refusé : l'accès demandé à la ressource est refusé.	

Ancien message d'erreur	Nouveau messages d'erreur	
CannotPullContainerError: Réponse d'erreur du démon : Get *imageURI* : net/http : demande annulée en attendant la connexion	CannotPullContainerError: La tâche ne peut pas extraire l'image. Vérifiez la configura tion de votre réseau. Réponse d'erreur du démon : Get : net/http *image* : demande annulée en attendant la connexion (Client.Timeout dépassé pendant l'attente des en-têtes)	
CannotPullContainerError<ip><port>: ref pull a été réessayé 5 fois : échec de la copie : échec de l'ouverture httpReadSeeker : échec de la demande : Get : dial tcp *registry-uri* : timeout i/o	CannotPullContainerError: La tâche ne peut pas être *image-uri* extraite du registre*registry-uri* . Il existe un problème de connexion entre la tâche et le registre. <port>Vérifiez la configuration de votre réseau de tâches. : échec de la copie httpReadSeeker : échec de l'ouverture : échec de la demande : Get *registry-uri* : dial tcp <ip>: timeout i/o	

ResourceNotFoundException

Ancien message d'erreur	Nouveau messages d'erreur	
Extraction de données secrètes depuis AWS Secrets Manager la *region* région : secret *sercretARN* :	ResourceNotFoundEx ception: La tâche ne peut pas récupérer le secret dont l'ARN est « *sercretARN* »	

Ancien message d'erreur	Nouveau messages d’erreur	
ResourceNotFoundException : Secrets Manager ne trouve pas le secret spécifié.	à partir de AWS Secrets Manager. Vérifiez si le secret existe dans la région spécifiée . ResourceNotFoundEx ception: Récupération de données secrètes depuis AWS Secrets Manager la région *region* : secret *sercretARN* : ResourceNotFoundEx ception : Secrets Manager ne trouve pas le secret spécifié.	

ResourceInitializationError

Ancien message d'erreur	Nouveau messages d’erreur	
ResourceInitializationError <ip><port>: impossible d'extraire les secrets ou l'authentification du registre : échec de la récupération de la ressource d'exécuti on : impossible de récupérer l'authentification du registre ECR : l'appel de service a été retenté 3 fois : : échec de la demande d'envoi dû à RequestError : Post «https://api.ecr.us-east-1.amazonaws.com/» : dial tcp : : timeout. i/o Vérifiez la configuration de votre réseau de tâches.	ResourceInitializationError : impossible d'extraire les secrets ou l'authentification du registre : la tâche ne peut pas extraire l'authentification du registre depuis Amazon ECR : il y a un problème de connexion entre la tâche et Amazon ECR. Vérifiez la configuration de votre réseau de tâches. RequestError: échec de la demande d'envoi dû à : Post "https://api.ecr.us-east-1.amazonaws.com« : dial tcp <ip>: <port>: timeout i/o	

Ancien message d'erreur	Nouveau messages d’erreur	
ResourceInitializationError : impossible d'extraire les secrets ou l'authentification du registre : échec de la récupération des ressource s d'exécution : impossible de récupérer les secrets de ssm : l'appel de service a été retenté 5 fois : : le contexte de la demande a été annulé en raison du RequestCanceled dépassement de la date limite du contexte	ResourceInitializationError : impossible d'extraire les secrets ou l'authentification du registre : impossible de récupérer les secrets de SSM : la tâche ne peut pas extraire les secrets de. AWS Systems Manager Il existe un problème de connexion entre la tâche et AWS Systems Manager Parameter Store. Vérifiez la configuration de votre réseau de tâches. RequestCanceled: contexte de demande annulé en raison de : date limite de contexte dépassée	
ResourceInitializationError: échec du téléchargement des fichiers env : commande de téléchargement de fichier : flux d'erreur non vide : RequestCanceled : contexte de demande annulé en raison du dépassement de la date limite du contexte	ResourceInitializationError: échec du téléchargement des fichiers env : la tâche ne peut pas télécharger les fichiers de variables d'environnement depuis Amazon S3. Il existe un problème de connexion entre la tâche et Amazon S3. Vérifiez la configuration de votre réseau de tâches. L'appel de service a été réessayé 5 fois : RequestCa nceled : le contexte de la demande a été annulé en raison du dépassement de la date limite du contexte	

Ancien message d'erreur	Nouveau messages d’erreur	
ResourceInitializationError : échec de la validation du logger args : :signal:killed	ResourceInitializationError : échec de la validation des arguments de l'enregis treur : la tâche ne trouve pas le groupe de CloudWatc h journaux Amazon défini dans la définition de la tâche. Il existe un problème de connexion entre la tâche et Amazon CloudWatch. Vérifiez la configuration de votre réseau. : signal : tué	
ResourceInitializationError : impossible d'extraire les secrets ou l'authentification du registre : échec de la commande pull : signal : tué	ResourceInitializationError : impossible d'extraire les secrets ou l'authentification du registre : vérifiez la configura tion de votre réseau de tâches. : signal : tué	

L'affichage d'Amazon ECS a permis de stopper les erreurs de tâches

Si vous avez des difficultés à démarrer une tâche, cette dernière peut avoir été arrêtée en raison d'une application ou d'erreurs de configuration. Par exemple, vous exécutez la tâche et elle affiche un statut `PENDING`, puis disparaît.

Si votre tâche a été créée par un service Amazon ECS, les actions entreprises par Amazon ECS pour gérer le service sont publiées dans les événements du service. Vous pouvez consulter les événements dans AWS Management Console AWS CLI l' AWS SDKsAPI Amazon ECS ou dans les outils utilisant l'API SDKs et. Ces événements incluent l'arrêt et le remplacement d'une tâche par Amazon ECS parce que les conteneurs de la tâche ont cessé de s'exécuter ou parce qu'un trop grand nombre de surveillances de l'état effectuées par Elastic Load Balancing ont échoué.

Si votre tâche s'est exécutée sur une instance de conteneur sur Amazon EC2 ou sur des ordinateurs externes, vous pouvez également consulter les journaux du runtime du conteneur et de l'agent

Amazon ECS. Ces journaux se trouvent sur l' EC2 instance Amazon hôte ou sur un ordinateur externe. Pour de plus amples informations, veuillez consulter Afficher les journaux des agents de conteneurs Amazon ECS.

Procédure

Console

AWS Management Console

Les étapes suivantes peuvent être utilisées pour vérifier la présence d'erreurs dans les tâches arrêtées à l'aide de la console. Pour voir les tâches arrêtées, vous devez modifier l'option de filtre.

Les tâches arrêtées n'apparaissent dans la console que pendant 1 heure.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur la *name* page Cluster :, choisissez l'onglet Tâches.
5. Configurez le filtre pour afficher les tâches arrêtées. Pour le statut souhaité du filtre, choisissez Arrêté.

   L'option Arrêté affiche vos tâches arrêtées et l'option Tout statut souhaité affiche toutes vos tâches.
6. Choisissez la tâche arrêtée à inspecter.
7. Dans la ligne correspondant à votre tâche arrêtée, dans la colonne Dernier statut, choisissez Arrêté.

   Une fenêtre contextuelle affiche le motif de l'arrêt.

AWS CLI

1. Répertoriez les tâches arrêtées d'un cluster. La sortie contient l'Amazon Resource Name (ARN) de la tâche, dont vous avez besoin pour décrire la tâche.

```
aws ecs list-tasks \
     --cluster cluster_name \
     --desired-status STOPPED \
```

```
     --region region
```

2. Décrivez la tâche arrêtée pour récupérer les informations. Pour plus d'informations, voir describe-tasks dans la AWS Command Line Interface référence.

```
aws ecs describe-tasks \
     --cluster cluster_name \
     --tasks arn:aws:ecs:region:account_id:task/cluster_name/task_ID \
     --region region
```

Utilisez les paramètres de sortie suivants.

- `stopCode`- Le code d'arrêt indique pourquoi une tâche a été arrêtée, par exemple ResourceInitializationError
- `StoppedReason`- La raison pour laquelle la tâche s'est arrêtée.
- `reason`(dans la `containers` structure) - La raison fournit des détails supplémentaires sur le conteneur arrêté.

Étapes suivantes

Consultez les tâches que vous avez arrêtées afin d'obtenir des informations sur leur cause. Pour de plus amples informations, veuillez consulter Messages d'erreur relatifs aux tâches interrompues par Amazon ECS.

Messages d'erreur relatifs aux tâches interrompues par Amazon ECS

Les messages d'erreur que vous pouvez recevoir lorsque votre tâche s'arrête de façon inattendue sont les suivants.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

Les codes d'erreur des tâches arrêtées sont associés à une catégorie, par exemple « ResourceInitializationError ». Pour obtenir plus d'informations sur chaque catégorie, consultez les rubriques suivantes :

Catégorie	En savoir plus
TaskFailedToStart	Résolution des TaskFailedToStart erreurs Amazon ECS
ResourceInitializationError	Résolution des ResourceInitializationError erreurs Amazon ECS
ResourceNotFoundException	Résolution des ResourceNotFoundException erreurs Amazon ECS
SpotInterruptionError	Résolution des SpotInterruption erreurs Amazon ECS
InternalError	Résolution des InternalError erreurs Amazon ECS
OutOfMemoryError	Résolution des OutOfMemoryError erreurs Amazon ECS
ContainerRuntimeError	Résolution des ContainerRuntimeError erreurs Amazon ECS
ContainerRuntimeTimeoutError	Résolution des ContainerRuntimeTimeoutError erreurs Amazon ECS
CannotStartContainerError	Résolution des CannotStartContainerError erreurs Amazon ECS
CannotStopContainerError	Résolution des CannotStopContainerError erreurs Amazon ECS

Catégorie	En savoir plus
CannotInspectContainerError	Résolution des CannotInspectContainerError erreurs Amazon ECS
CannotCreateVolumeError	Résolution des CannotCreateVolumeError erreurs Amazon ECS
CannotPullContainer	CannotPullContainer erreurs de tâche dans Amazon ECS

Résolution des TaskFailedToStart erreurs Amazon ECS

Vous trouverez ci-dessous des messages `TaskFailedToStart` d'erreur et des actions que vous pouvez entreprendre pour corriger les erreurs.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

EC2 Erreur inattendue lors de la tentative de création d'une interface réseau avec l'attribution d'adresses IP publiques activée dans le sous-réseau '***subnet-id***

Cela se produit lorsqu'une tâche Fargate utilisant `awsvpc` le mode réseau s'exécute dans un sous-réseau doté d'une adresse IP publique et que le sous-réseau ne possède pas suffisamment d'adresses IP.

Le nombre d'adresses IP disponibles est disponible sur la page des détails du sous-réseau de la EC2 console Amazon, ou en utilisant`describe-subnets`. Pour plus d'informations, consultez Afficher votre sous-réseau dans le guide de l'utilisateur Amazon VPC.

Pour résoudre ce problème, vous pouvez créer un nouveau sous-réseau dans lequel exécuter votre tâche.

InternalError: ***<reason>***

Cette erreur se produit lorsqu'une pièce jointe ENI est demandée. Amazon gère le provisionnement de l'ENI de EC2 manière asynchrone. Le processus de provisionnement prend du temps.

Amazon ECS dispose d'un délai d'expiration au cas où les temps d'attente seraient longs ou les défaillance non signalées. Parfois, l'ENI est provisionnée, mais le rapport arrive à Amazon ECS après l'expiration du délai de défaillance. Dans ce cas, Amazon ECS détecte la défaillance de la tâche signalée avec un ENI en cours d'utilisation.

La définition de tâche sélectionnée n'est pas compatible avec la stratégie de calcul sélectionnée

Cette erreur se produit lorsque vous avez choisi une définition de tâche dont le type de lancement ne correspond pas au type de capacité du cluster. Pour de plus amples informations, veuillez consulter Types de lancement Amazon ECS. Vous devez sélectionner une définition de tâche qui correspond au fournisseur de capacité attribué à votre cluster.

Impossible de connecter l'interface réseau à l'index des appareils non utilisés

Cette erreur se produit lorsque vous utilisez le type `awsvpc` de réseau et qu'il n'y en a pas assez CPU/memory pour la tâche. Tout d'abord, vérifiez la présence des instances sur le processeur. Pour plus d'informations, consultez les spécifications relatives aux types d' EC2 instances Amazon dans la section Types d' EC2 instances Amazon. Prenez la valeur du processeur pour l'instance et multipliez-la par le nombre de ENIs pour l'instance. Utilisez cette valeur e dans la définition de la tâche.

AGENT

L'instance de conteneur avec laquelle vous avez essayé de lancer une tâche comporte un agent qui est actuellement déconnecté. Afin d'éviter de longs délais d'attente pour le placement de la tâche, la demande a été rejetée.

Pour plus d'informations sur la résolution problèmes des agents déconnectés, consultez Comment résoudre les problèmes liés à un agent Amazon ECS déconnecté ?.

Résolution des ResourceInitializationError erreurs Amazon ECS

Vous trouverez ci-dessous des messages `ResourceInitialization` d'erreur et des actions que vous pouvez entreprendre pour corriger les erreurs.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

Erreurs

- La tâche ne peut pas extraire l'authentification du registre depuis Amazon ECR. Il existe un problème de connexion entre la tâche et Amazon ECR. Vérifiez la configuration de votre réseau de tâches.

- La tâche ne peut pas télécharger les fichiers de variables d'environnement depuis Amazon S3. Il existe un problème de connexion entre la tâche et Amazon S3. Vérifiez la configuration de votre réseau de tâches.
- La tâche ne peut pas extraire les secrets du AWS Systems Manager Parameter Store. Vérifiez votre connexion réseau entre la tâche et AWS Systems Manager.
- La tâche ne peut pas en extraire des secrets AWS Secrets Manager. Il existe un problème de connexion entre la tâche et Secrets Manager. Vérifiez la configuration de votre réseau de tâches.
- La tâche ne peut pas extraire le secret de Secrets Manager. La tâche ne peut pas récupérer le secret avec l'ARN « secretARN » dans Secrets Manager. Vérifiez si le secret existe dans la région spécifiée.
- échec de la commande pull : impossible d'extraire les secrets ou l'authentification du registre Vérifiez la configuration de votre réseau de tâches.
- La tâche ne trouve pas le groupe de CloudWatch journaux Amazon défini dans la définition de la tâche. Il existe un problème de connexion entre la tâche et Amazon CloudWatch. Vérifiez la configuration de votre réseau.
- échec de l'initialisation du pilote de journalisation
- Impossible d'invoquer les commandes EFS utils pour configurer les volumes EFS

La tâche ne peut pas extraire l'authentification du registre depuis Amazon ECR. Il existe un problème de connexion entre la tâche et Amazon ECR. Vérifiez la configuration de votre réseau de tâches.

Cette erreur indique que la tâche ne peut pas se connecter à Amazon ECR.

Vérifiez la connexion entre la tâche et Amazon ECR. Pour plus d'informations, consultez Vérification de l'arrêt de la connectivité des tâches par Amazon ECS.

La tâche ne peut pas télécharger les fichiers de variables d'environnement depuis Amazon S3. Il existe un problème de connexion entre la tâche et Amazon S3. Vérifiez la configuration de votre réseau de tâches.

Cette erreur se produit lorsque votre tâche ne parvient pas à télécharger votre fichier d'environnement depuis Amazon S3.

Vérifiez la connexion entre la tâche et le point de terminaison Amazon S3 VPC. Pour plus d'informations, consultez Vérification de l'arrêt de la connectivité des tâches par Amazon ECS.

La tâche ne peut pas extraire les secrets du AWS Systems Manager Parameter Store. Vérifiez votre connexion réseau entre la tâche et AWS Systems Manager.

Cette erreur se produit lorsque votre tâche ne parvient pas à extraire l'image définie dans la définition de tâche à l'aide des informations d'identification dans Systems Manager.

Vérifiez la connexion entre la tâche et le point de terminaison VPC de Systems Manager. Pour plus d'informations, consultez Vérification de l'arrêt de la connectivité des tâches par Amazon ECS.

La tâche ne peut pas en extraire des secrets AWS Secrets Manager. Il existe un problème de connexion entre la tâche et Secrets Manager. Vérifiez la configuration de votre réseau de tâches.

Cette erreur se produit lorsque votre tâche ne parvient pas à extraire l'image définie dans la définition de la tâche à l'aide des informations d'identification dans Secrets Manager.

L'erreur indique qu'il existe un problème de connectivité réseau entre le point de terminaison VPC de Systems Manager et la tâche.

Pour plus d'informations sur la façon de vérifier la connectivité entre la tâche et le point de terminaison, consultezVérification de l'arrêt de la connectivité des tâches par Amazon ECS.

La tâche ne peut pas extraire le secret de Secrets Manager. La tâche ne peut pas récupérer le secret avec l'ARN « ***secretARN*** » dans Secrets Manager. Vérifiez si le secret existe dans la région spécifiée.

Cette erreur se produit lorsque votre tâche ne parvient pas à extraire l'image définie dans la définition de la tâche à l'aide des informations d'identification dans Secrets Manager.

Ce problème est dû à l'une des raisons suivantes :

Cause de l'erreur.	Faites ceci...	
Problème de connectivité réseau entre le point de terminaison VPC Secrets Manager et la tâche. Le problème est lié au réseau lorsque l'une des chaînes suivantes apparaît dans le message d'erreur :	Vérifiez la connectivité entre la tâche et le point de terminais on Secrets Manager. Pour de plus amples informations, veuillez consulter Vérification de l'arrêt de la connectivité des tâches par Amazon ECS.	

Cause de l'erreur.	Faites ceci...	
• cadran TCP • appel UDP • <ip>: <port>: i/o délai d'expiration • net/http : délai d'expiration de la poignée de main TLS • lire : le délai de connexion a expiré • Client.Timeout dépassé pendant l'attente des en-têtes • net/http : demande annulée en attendant la connexion • signal : tué • date limite de contexte dépassée		
Le rôle d'exécution de la tâche défini dans la définition de la tâche ne dispose pas des autorisations nécessaires pour Secrets Manager.	Ajoutez les autorisations requises pour Secrets Manager au rôle d'exécuti on des tâches. Pour de plus amples informations, veuillez consulter Permissions de Secrets Manager ou de Systems Manager.	

Cause de l'erreur.	Faites ceci...	
L'ARN secret n'existe pas	Vérifiez que l'ARN existe dans Secrets Manager. Pour plus d'informations sur l'affiche de vos images, consultez la section Rechercher des secrets dans Secrets Manager du Guide du développeur de Secrets Manager.	

échec de la commande pull : impossible d'extraire les secrets ou l'authentification du registre Vérifiez la configuration de votre réseau de tâches.

Cette erreur se produit lorsque votre tâche ne parvient pas à se connecter à Amazon ECR, Systems Manager ou Secrets Manager. Cela est dû à une mauvaise configuration de votre réseau.

Pour résoudre ce problème, vérifiez la connectivité entre la tâche et Amazon ECR. Vous devez également vérifier la connectivité entre votre tâche et le service qui stocke votre secret (Systems Manager ou Secrets Manager). Pour de plus amples informations, veuillez consulter Vérification de l'arrêt de la connectivité des tâches par Amazon ECS.

La tâche ne trouve pas le groupe de CloudWatch journaux Amazon défini dans la définition de la tâche. Il existe un problème de connexion entre la tâche et Amazon CloudWatch. Vérifiez la configuration de votre réseau.

Cette erreur se produit lorsque votre tâche ne trouve pas le groupe de CloudWatch journaux que vous avez défini dans la définition de la tâche.

L'erreur indique qu'il existe un problème de connectivité réseau entre le point de terminaison du CloudWatch VPC et la tâche.

Pour plus d'informations sur la façon de vérifier la connectivité entre la tâche et le point de terminaison, consultezVérification de l'arrêt de la connectivité des tâches par Amazon ECS.

échec de l'initialisation du pilote de journalisation

Cette erreur se produit lorsque votre tâche ne trouve pas le groupe de CloudWatch journaux que vous avez défini dans la définition de la tâche.

L'erreur indique que le CloudWatch groupe dans la définition de tâche n'existe pas.

Suivez les étapes ci-dessous pour trouver les éléments manquants CloudWatch.

1. Exécutez la commande suivante pour obtenir les informations de définition de tâche.

```
aws ecs describe-task-definition \
    --task-definition task-def-name
```

   Examinez le résultat de chaque conteneur et notez la `awslogs-group` valeur.

```
"logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/example-group",
                    "awslogs-create-group": "true",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "ecs"
                },
```

2. Vérifiez que le groupe existe. CloudWatch Pour plus d'informations, consultez la section Utilisation des groupes de journaux et des flux de journaux dans le guide de l'utilisateur Amazon CloudWatch Logs.

   Le problème est soit que le groupe spécifié dans la définition de la tâche est incorrect, soit que le groupe de journaux n'existe pas.

3. Résolvez le problème.

Le problème est que...	Faites ceci...
Le groupe de journaux spécifié dans la définition de la tâche est incorrect.	Mettez à jour la définitio n de tâche pour inclure la configuration du groupe de journaux dans la définitio n du conteneur. Pour plus d'informations sur la mise à jour de la définition de tâche, consultez Mettre à jour une définition de tâche Amazon

Le problème est que...	Faites ceci...	
	ECS à l'aide de la console ou RegisterTaskDefinitionconsultez le manuel Amazon Elastic Container Service API Reference.	
Le groupe de journaux n'existe pas dans CloudWatch	Créez le groupe de journaux. Pour plus d'informations, consultez la section Créer un groupe de CloudWatch journaux dans Logs du guide de l'utilisateur Amazon CloudWatch Logs.	

Impossible d'invoquer les commandes EFS utils pour configurer les volumes EFS

Les problèmes suivants peuvent vous empêcher de monter vos volumes Amazon EFS selon vos demandes :

- Le système de fichiers Amazon EFS n'est pas configuré correctement.
- La tâche ne dispose pas des autorisations requises.
- Il existe des problèmes liés aux configurations réseau et VPC.

Pour plus d'informations sur la façon de déboguer et de résoudre ce problème, consultez Pourquoi ne puis-je pas monter mes volumes Amazon EFS sur mes AWS Fargate tâches sur AWS Re:post.

Résolution des ResourceNotFoundException erreurs Amazon ECS

Vous trouverez ci-dessous des messages `ResourceNotFoundException` d'erreur et des actions que vous pouvez entreprendre pour corriger les erreurs.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

La tâche ne peut pas récupérer le secret dont l'ARN est ***sercretARN*** « » de AWS Secrets Manager. Vérifiez si le secret existe dans la région spécifiée.

Cette erreur se produit lorsque la tâche ne parvient pas à récupérer le secret depuis Secrets Manager. Cela signifie que le secret spécifié dans la définition de la tâche (et contenu dans le message d'erreur) n'existe pas dans Secrets Manager.

La région figure dans le message d'erreur.

Extraction de données secrètes depuis AWS Secrets Manager la région *region* : secret *sercretARN* : ResourceNotFoundException : Secrets Manager ne trouve pas le secret spécifié.

Pour plus d'informations sur la recherche d'un secret, voir Rechercher des secrets AWS Secrets Manager dans le guide de AWS Secrets Manager l'utilisateur.

Utilisez le tableau suivant pour déterminer et corriger l'erreur.

Problème	Actions
Le secret se trouve dans une région différente de celle de la définition de la tâche.	a. Créez le secret dans la même région que la tâche. Pour plus d'informations, voir Création d'un AWS Secrets Manager secret. b. Mettez à jour la définition de la tâche avec le nouveau secret. Pour plus d'informations, consultez Mettre à jour une définition de tâche Amazon ECS à l'aide de la console ou consultez le RegisterTaskDefinitionmanuel Amazon Elastic Container Service API Reference.
L'ARN secret de la définition de tâche est incorrect. Le	Mettez à jour la définition de tâche avec le secret correct.

Problème	Actions
secret correct existe dans Secrets Manager.	Pour plus d'informations, consultez Mettre à jour une définition de tâche Amazon ECS à l'aide de la console ou consultez le RegisterTaskDefinitionmanuel Amazon Elastic Container Service API Reference.
Le secret n'existe plus.	a. Créez le secret dans la même région que la tâche. Pour plus d'informations, voir Création d'un AWS Secrets Manager secret. b. Mettez à jour la définition de la tâche avec le nouveau secret. Pour plus d'informations, consultez Mettre à jour une définition de tâche Amazon ECS à l'aide de la console ou consultez le RegisterTaskDefinitionmanuel Amazon Elastic Container Service API Reference.

Résolution des SpotInterruption erreurs Amazon ECS

L'`SpotInterruption`erreur a différentes causes en fonction du type Fargate EC2 et du type de lancement.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

Type de lancement Fargate

L'`SpotInterruption`erreur se produit lorsqu'il n'y a pas de capacité Fargate Spot ou lorsque Fargate reprend la capacité de Fargate Spot.

Vous pouvez exécuter vos tâches dans plusieurs zones de disponibilité pour augmenter la capacité.

EC2 type de lancement

Cette erreur se produit lorsqu'aucune instance Spot n'est disponible ou EC2 lorsque la capacité d'une instance Spot est rétablie.

Vous pouvez exécuter vos instances dans plusieurs zones de disponibilité pour augmenter la capacité.

Résolution des InternalError erreurs Amazon ECS

S'applique à : type de lancement Fargate

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

`InternalError`Erreur lorsque l'agent rencontre une erreur interne inattendue non liée à l'exécution.

Cette erreur se produit uniquement si vous utilisez la version `1.4` ou ultérieure de la plateforme.

Pour plus d'informations sur la façon de déboguer et de résoudre ce problème, consultezMessages d'erreur relatifs aux tâches interrompues par Amazon ECS.

Résolution des OutOfMemoryError erreurs Amazon ECS

Vous trouverez ci-dessous des messages OutOfMemoryError d'erreur et des actions que vous pouvez entreprendre pour corriger les erreurs.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

conteneur détruit en raison de l'utilisation de la mémoire

Cette erreur se produit lorsqu'un conteneur se ferme parce que les processus du conteneur consomment plus de mémoire que celle allouée dans la définition de la tâche, ou en raison de contraintes liées à l'hôte ou au système d'exploitation.

Résolution des ContainerRuntimeError erreurs Amazon ECS

Vous trouverez ci-dessous des messages ContainerRuntimeError d'erreur et des actions que vous pouvez entreprendre pour corriger les erreurs.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

ContainerRuntimeError

Cette erreur se produit lorsque l'agent reçoit une erreur inattendue de `containerd` pour une opération spécifique à l'exécution. Cette erreur est généralement causée par une défaillance interne de l'agent ou de l'environnement d'exécution `containerd`.

Cette erreur se produit uniquement si vous utilisez la version `1.4.0` ou ultérieure de la plateforme (Linux) ou `1.0.0`ou une version ultérieure (Windows).

Pour plus d'informations sur la façon de déboguer et de résoudre ce problème, consultez Pourquoi ma tâche Amazon ECS est-elle arrêtée sur AWS Re:Post.

Résolution des ContainerRuntimeTimeoutError erreurs Amazon ECS

Vous trouverez ci-dessous des messages ContainerRuntimeTimeoutError d'erreur et des actions que vous pouvez entreprendre pour corriger les erreurs.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

Impossible de passer à l'exécution ; le délai a expiré après 1 m d'attente ou erreur de temporisation dans Docker

Cette erreur se produit lorsqu'un conteneur n'a pas pu passer à l'état `RUNNING` ou `STOPPED` dans le délai d'expiration. La raison et la valeur du délai d'expiration sont fournies dans le message d'erreur.

Résolution des CannotStartContainerError erreurs Amazon ECS

Vous trouverez ci-dessous des messages CannotStartContainerError d'erreur et des actions que vous pouvez entreprendre pour corriger les erreurs.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

Impossible d'obtenir le statut du conteneur : ***<reason>***

Cette erreur se produit lorsqu'un conteneur ne peut pas être démarré.

Si votre conteneur tente de dépasser la mémoire spécifiée ici, il est arrêté. Augmentez la mémoire présentée au conteneur. Il s'agit du `memory` paramètre figurant dans la définition de la tâche. Pour de plus amples informations, veuillez consulter the section called “Mémoire”.

Résolution des CannotStopContainerError erreurs Amazon ECS

Vous trouverez ci-dessous des messages CannotStopContainerError d'erreur et des actions que vous pouvez entreprendre pour corriger les erreurs.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

CannotStopContainerError

Cette erreur se produit lorsqu'un conteneur ne peut pas être arrêté.

Pour plus d'informations sur la façon de déboguer et de résoudre ce problème, consultez Pourquoi ma tâche Amazon ECS est-elle arrêtée sur AWS Re:Post.

Résolution des CannotInspectContainerError erreurs Amazon ECS

Vous trouverez ci-dessous des messages CannotInspectContainerError d'erreur et des actions que vous pouvez entreprendre pour corriger les erreurs.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

CannotInspectContainerError

Cette erreur se produit lorsque l'agent de conteneur ne peut pas décrire le conteneur via son environnement d'exécution.

Lorsque vous utilisez une version de plate-forme `1.3` ou une version antérieure, l'agent Amazon ECS renvoie la raison à partir de Docker.

Lors de l'utilisation d'`1.4.0`une version de plate-forme ou ultérieure (Linux) `1.0.0` ou ultérieure (Windows), l'agent Fargate renvoie la raison de. `containerd`

Pour plus d'informations sur la façon de déboguer et de résoudre ce problème, consultez Pourquoi ma tâche Amazon ECS est-elle arrêtée sur AWS Re:Post.

Résolution des CannotCreateVolumeError erreurs Amazon ECS

Vous trouverez ci-dessous des messages CannotCreateVolumeError d'erreur et des actions que vous pouvez entreprendre pour corriger les erreurs.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

CannotCreateVolumeError

Cette erreur se produit lorsque l'agent ne parvient pas à créer le montage de volume spécifié dans la définition de tâche.

Cette erreur se produit uniquement si vous utilisez la version `1.4.0` ou ultérieure de la plateforme (Linux) ou `1.0.0`ou une version ultérieure (Windows).

Pour plus d'informations sur la façon de déboguer et de résoudre ce problème, consultez Pourquoi ma tâche Amazon ECS est-elle arrêtée sur AWS Re:Post.

CannotPullContainer erreurs de tâche dans Amazon ECS

Les erreurs suivantes indiquent que la tâche n'a pas pu démarrer car Amazon ECS ne parvient pas à récupérer l'image de conteneur spécifiée.

> **Note**
>
> La version 1.4 de la plateforme Fargate tronque les messages d'erreur longs.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

Erreurs

- La tâche ne peut pas extraire l'image. Vérifiez que le rôle dispose des autorisations nécessaires pour extraire des images du registre.
- La tâche ne peut pas extraire image-name « » du référentiel Amazon ECR « repository URI ». Il existe un problème de connexion entre la tâche et Amazon ECR. Vérifiez la configuration de votre réseau de tâches.
- La tâche ne peut pas extraire l'image. Vérifiez la configuration de votre réseau

- CannotPullContainerError: le manifeste pull image a été réessayé 5 fois : échec de la résolution de la référence
- Erreur API (500) : demande Get https://111122223333.dkr.ecr.us-east-1.amazonaws.com/v2/ : net/http : annulée en attendant la connexion
- Erreur d'API
- write/var/lib/docker/tmp/GetImageBlob111111111: il ne reste plus d'espace sur l'appareil
- ERREUR : toomanyrequests : trop de demandes ou vous avez atteint votre limite de taux d'extraction.
- Réponse d'erreur du démon : Get url : net/http : demande annulée en attendant la connexion
- ref pull a été réessayé 1 fois : échec de la copie : échec de l'ouverture httpReaderSeeker : code d'état inattendu
- accès au pull refusé
- échec de la commande pull : panique : erreur d'exécution : adresse mémoire non valide ou déréférencement du pointeur nul
- erreur lors de l' conf/error extraction de l'image lors de la configuration
- Contexte annulé

La tâche ne peut pas extraire l'image. Vérifiez que le rôle dispose des autorisations nécessaires pour extraire des images du registre.

Cette erreur indique que la tâche ne peut pas extraire l'image spécifiée dans la définition de la tâche en raison de problèmes d'autorisation.

Pour résoudre ce problème :

1. Vérifiez que l'image existe dans le`repository`. Pour plus d'informations sur l'affichage de vos images, consultez la section Affichage des détails des images dans Amazon ECR dans le guide de l'utilisateur d'Amazon Elastic Container Registry.
2. Vérifiez que vous disposez des autorisations appropriées pour extraire l'image. `role-arn`

   Pour plus d'informations sur la mise à jour des rôles, consultez la section Mettre à jour les autorisations pour un rôle dans le guide AWS Identity and Access Management d'utilisation.

   La tâche utilise l'un des rôles suivants :

   - Pour les tâches de type de lancement Fargate, il s'agit du rôle d'exécution des tâches. Pour plus d'informations sur les autorisations supplémentaires pour Amazon ECR,Tâches Fargate

extrayant des images Amazon ECR au-delà des autorisations des points de terminaison de l'interface.

- Pour les tâches de type EC2 lancement, il s'agit du rôle d'instance de conteneur. Pour plus d'informations sur les autorisations supplémentaires pour Amazon ECR,Autorisations Amazon ECR.

La tâche ne peut pas extraire ***image-name*** « » du référentiel Amazon ECR « ***repository URI*** ». Il existe un problème de connexion entre la tâche et Amazon ECR. Vérifiez la configuration de votre réseau de tâches.

Cette erreur indique que la tâche ne peut pas se connecter à Amazon ECR. Vérifiez la connexion au *repository URI* référentiel.

Pour plus d'informations sur la façon de vérifier et de résoudre le problème, consultezVérification de l'arrêt de la connectivité des tâches par Amazon ECS.

La tâche ne peut pas extraire l'image. Vérifiez la configuration de votre réseau

Cette erreur indique que la tâche ne peut pas se connecter à Amazon ECR.

Pour plus d'informations sur la façon de vérifier et de résoudre le problème, consultezVérification de l'arrêt de la connectivité des tâches par Amazon ECS.

CannotPullContainerError: le manifeste pull image a été réessayé 5 fois : échec de la résolution de la référence

Cette erreur indique que la tâche ne peut pas extraire l'image.

Pour résoudre cela, vous pouvez :

- Vérifiez que l'image spécifiée dans la définition de tâche correspond à l'image du référentiel.
- Amazon ECS impose la stabilité de la version de l'image. Si l'image d'origine n'est plus disponible, cette erreur s'affiche. La balise image permet de renforcer ce comportement. Changez l'image dans la définition de la tâche en utilisant : latest comme balise par une version spécifique. Pour de plus amples informations, veuillez consulter Résolution de l'image du conteneur.

Pour plus d'informations sur la façon de vérifier et de résoudre le problème, consultezVérification de l'arrêt de la connectivité des tâches par Amazon ECS.

Erreur API (500) : demande Get https://111122223333.dkr.ecr.us-east-1.amazonaws.com/v2/ : net/http : annulée en attendant la connexion

Cette erreur indique qu'une connexion a expiré, car il n'existe aucune route vers Internet.

Pour résoudre ce problème, vous pouvez :

- Pour des tâches de sous-réseaux publics, spécifiez ENABLED (Activé) pour Auto-assign public IP (Attribuer automatiquement l'adresse IP publique) lors du lancement de la tâche. Pour de plus amples informations, veuillez consulter Exécution d'une application en tant que tâche Amazon ECS.
- Pour des tâches de sous-réseaux privés, spécifiez Désactivé pour Attribuer automatiquement l'adresse IP publique lors du lancement de la tâche, puis configurez une passerelle NAT dans votre VPC pour acheminer les demandes vers Internet. Pour plus d'informations, consultez Passerelles NAT dans le Guide de l'utilisateur Amazon VPC.

Erreur d'API

Cette erreur indique qu'il existe un problème de connexion avec le point de terminaison Amazon ECR.

Pour plus d'informations sur la manière de résoudre ce problème, consultez l'article Comment puis-je résoudre l'erreur Amazon ECR « CannotPullContainerError : erreur d'API » dans Amazon ECS sur le Support site Web.

write/var/lib/docker/tmp/***GetImageBlob111111111***: il ne reste plus d'espace sur l'appareil

Cette erreur indique que l'espace disque est insuffisant.

Pour résoudre ce problème, vous devez libérer de l'espace disque.

Si vous utilisez l'AMI optimisée pour Amazon ECS, vous pouvez utiliser la commande suivante pour récupérer les 20 fichiers les plus volumineux de votre système de fichiers :

```
du -Sh / | sort -rh | head -20
```

Exemple de sortie :

```
5.7G    /var/lib/docker/
containers/50501b5f4cbf90b406e0ca60bf4e6d4ec8f773a6c1d2b451ed8e0195418ad0d2
```

```
1.2G    /var/log/ecs
594M    /var/lib/docker/devicemapper/mnt/
c8e3010e36ce4c089bf286a623699f5233097ca126ebd5a700af023a5127633d/rootfs/data/logs
...
```

Dans certains cas, le volume racine peut être rempli par un conteneur en cours d'exécution. Si le conteneur utilise le pilote de journal `json-file` par défaut sans limite `max-size`, le fichier journal est peut-être responsable de la plupart de l'utilisation de cet espace. Vous pouvez utiliser la commande `docker ps` pour vérifier quel conteneur utilise l'espace en mappant le nom du répertoire dans la sortie ci-dessus à l'ID de conteneur. Par exemple :

```
CONTAINER ID   IMAGE                              COMMAND             CREATED
 STATUS              PORTS                              NAMES
50501b5f4cbf   amazon/amazon-ecs-agent:latest   "/agent"            4 days ago
 Up 4 days                                             ecs-agent
```

Par défaut, lorsque vous utilisez le pilote de journal `json-file`, Docker capture la sortie standard (et l'erreur standard) de tous vos conteneurs et les écrit dans les fichiers au format JSON. Vous pouvez définir l'option `max-size` comme une option de pilote de journal, ce qui empêche le fichier journal d'occuper trop d'espace. Pour plus d'informations, consultez le pilote de journalisation de fichiers JSON dans la documentation Docker.

Voici un extrait de définition de conteneur illustrant la manière d'utiliser cette option :

```
{
    "log-driver": "json-file",
    "log-opts": {
        "max-size": "256m"
    }
}
```

Si les journaux de vos conteneurs occupent trop d'espace disque, vous pouvez également utiliser le pilote de `awslogs` journal. Le pilote de `awslogs` journal envoie les journaux à CloudWatch, ce qui libère de l'espace disque qui serait autrement utilisé pour vos journaux de conteneur sur l'instance de conteneur. Pour de plus amples informations, veuillez consulter Envoyez les journaux Amazon ECS à CloudWatch .

Vous devrez peut-être mettre à jour la taille du disque auquel Docker peut accéder.

Pour plus d'informations, voir CannotPullContainerError: il ne reste plus d'espace sur l'appareil.

ERREUR : toomanyrequests : trop de demandes ou vous avez atteint votre limite de taux d'extraction.

Cette erreur indique qu'il existe une limite de débit pour Docker Hub.

Si vous recevez un message pour l'une des erreurs suivantes, vous êtes probablement en train d'atteindre les limites de débit Docker Hub :

Pour en savoir plus sur les limites de débit Docker Hub, consultez Understanding Docker Hub rate limiting.

Si vous avez augmenté la limite de débit de Docker Hub et que vous devez authentifier vos extractions Docker pour vos instances de conteneur, consultez Authentification de registre privé pour les instances de conteneur.

Réponse d'erreur du démon : Get ***url*** : net/http : demande annulée en attendant la connexion

Cette erreur indique qu'une connexion a expiré, car il n'existe aucune route vers Internet.

Pour résoudre ce problème, vous pouvez :

- Pour des tâches de sous-réseaux publics, spécifiez ENABLED (Activé) pour Auto-assign public IP (Attribuer automatiquement l'adresse IP publique) lors du lancement de la tâche. Pour de plus amples informations, veuillez consulter Exécution d'une application en tant que tâche Amazon ECS.
- Pour des tâches de sous-réseaux privés, spécifiez Désactivé pour Attribuer automatiquement l'adresse IP publique lors du lancement de la tâche, puis configurez une passerelle NAT dans votre VPC pour acheminer les demandes vers Internet. Pour plus d'informations, consultez Passerelles NAT dans le Guide de l'utilisateur Amazon VPC.

ref pull a été réessayé 1 fois : échec de la copie : échec de l'ouverture httpReaderSeeker : code d'état inattendu

Cette erreur indique qu'un échec s'est produit lors de la copie d'une image.

Pour résoudre ce problème, consultez l'un des articles suivants :

- Pour les tâches Fargate, consultez Comment puis-je corriger l'erreur « cannotpullcontainererror » pour mes tâches Amazon ECS sur Fargate ?.
- Pour les autres tâches, consultez Comment puis-je corriger l'erreur « cannotpullcontainererror » pour mes tâches Amazon ECS ?.

accès au pull refusé

Cette erreur indique qu'il n'est pas possible d'accéder à l'image.

Pour résoudre ce problème, vous devrez peut-être authentifier votre client Docker auprès d'Amazon ECR. Pour plus d'informations, consultez la section Authentification du registre privé dans le guide de l'utilisateur Amazon ECR.

échec de la commande pull : panique : erreur d'exécution : adresse mémoire non valide ou déréférencement du pointeur nul

Cette erreur indique qu'il n'est pas possible d'accéder à l'image en raison d'une adresse mémoire non valide ou d'un déréférencement nul du pointeur.

Pour résoudre ce problème :

- Vérifiez que vous disposez des règles du groupe de sécurité pour accéder à Amazon S3.
- Lorsque vous utilisez des points de terminaison de passerelle, vous devez ajouter une route dans la table de routage pour accéder au point de terminaison.

erreur lors de l' conf/error extraction de l'image lors de la configuration

Cette erreur indique qu'une limite de débit a été atteinte ou qu'il y a une erreur réseau :

Pour résoudre ce problème, consultez Comment puis-je résoudre l'erreur « CannotPullContainerError » dans ma tâche de type de EC2 lancement Amazon ECS.

Contexte annulé

Cette erreur indique que le contexte a été annulé.

La cause courante de cette erreur est que le VPC utilisé par votre tâche n'a pas de route pour extraire l'image d'Amazon ECR.

Vérification de l'arrêt de la connectivité des tâches par Amazon ECS

Il arrive qu'une tâche s'arrête en raison d'un problème de connectivité réseau. Il peut s'agir d'un problème intermittent, mais il est probablement dû au fait que la tâche ne peut pas se connecter à un point de terminaison.

Test de la connectivité des tâches

Vous pouvez utiliser `AWSSupport-TroubleshootECSTaskFailedToStart` runbook pour tester la connectivité des tâches. Lorsque vous utilisez le runbook, vous avez besoin des informations suivantes sur les ressources :

- L'ID de la tâche

  Utilisez l'ID de la dernière tâche ayant échoué.
- Le cluster dans lequel se trouvait la tâche

Pour plus d'informations sur l'utilisation du runbook, consultez la référence `AWSSupport-TroubleshootECSTaskFailedToStart`du runbook AWS Systems Manager Automation.

Le runbook analyse la tâche. Vous pouvez consulter les résultats dans la section Sortie pour les problèmes suivants susceptibles d'empêcher le démarrage d'une tâche :

- Connectivité réseau avec le registre de conteneurs configuré
- Connectivité des points de terminaison VPC
- Configuration des règles du groupe de sécurité

Résolution des problèmes liés aux terminaux VPC

Lorsque le résultat du `AWSSupport-TroubleshootECSTaskFailedToStart` runbook indique le problème du point de terminaison VPC, vérifiez la configuration suivante :

- Le VPC sur lequel vous créez le point de terminaison doit utiliser le DNS privé.
- Assurez-vous que vous disposez d'un AWS PrivateLink point de terminaison pour le service auquel la tâche ne peut pas se connecter dans le même VPC que la tâche. Pour plus d'informations, consultez l'un des sites suivants :

Service	Informations sur le point de terminaison VPC pour le service	
Amazon ECR	Points de terminaison VPC de l'interface Amazon ECR ()AWS PrivateLink	
Systems Manager	Améliorez la sécurité des EC2 instances en utilisant des points de terminaison VPC pour Systems Manager	
Secrets Manager	Utilisation d'un point de AWS Secrets Manager terminaison VPC	
CloudWatch	CloudWatch Point de terminaison d'un VPC	
Amazon S3	AWS PrivateLink pour Amazon S3	

- Configurez une règle de sortie pour le sous-réseau de tâches qui autorise le protocole HTTPS sur le trafic DNS (TCP) du port 443. Pour plus d'informations, consultez Configurer les règles des groupes de sécurité dans le guide de l'utilisateur d'Amazon Elastic Compute Cloud.
- Si vous utilisez un serveur de noms de domaine personnalisé, confirmez les paramètres de la requête DNS. La requête doit avoir un accès sortant sur le port 53 et utiliser les protocoles UDP et TCP. Il doit également disposer d'un accès HTTPS sur le port 443. Pour plus d'informations, consultez la section Configurer les règles des groupes de sécurité dans le guide de l'utilisateur d'Amazon Elastic Compute Cloud.
- Si le sous-réseau possède une ACL réseau, les règles ACL suivantes sont requises :
  - Règle sortante qui autorise le trafic qui autorise le trafic sur les ports 1024 à 65535.
  - Règle entrante qui autorise le trafic TCP sur le port 443.

Pour plus d'informations sur la configuration des règles, consultez la section Contrôler le trafic vers les sous-réseaux à l'aide du réseau ACLs dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.

Résoudre les problèmes de réseau

Lorsque le résultat du `AWSSupport-TroubleshootECSTaskFailedToStart` runbook indique un problème réseau, vérifiez la configuration suivante :

Tâches utilisant le mode réseau awsvpc dans un sous-réseau public

Effectuez la configuration suivante en fonction du runbook :

- Pour des tâches de sous-réseaux publics, spécifiez ENABLED (Activé) pour Auto-assign public IP (Attribuer automatiquement l'adresse IP publique) lors du lancement de la tâche. Pour de plus amples informations, veuillez consulter Exécution d'une application en tant que tâche Amazon ECS.
- Vous avez besoin d'une passerelle pour gérer le trafic Internet. La table de routage du sous-réseau de tâches doit comporter un itinéraire pour le trafic vers la passerelle.

  Pour plus d'informations, consultez la section Ajouter et supprimer des itinéraires d'une table de routage dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.

Type de passerelle	Destination de la table de routage	Cible de la table de routage
NAT	0.0.0.0/0	ID de passerelle NAT
Passerelle Internet	0.0.0.0/0	ID de passerelle Internet

- Si le sous-réseau de tâches possède une ACL réseau, les règles ACL suivantes sont requises :
  - Règle sortante qui autorise le trafic sur les ports 1024 à 65535.
  - Règle entrante qui autorise le trafic TCP sur le port 443.

  Pour plus d'informations sur la configuration des règles, consultez la section Contrôler le trafic vers les sous-réseaux à l'aide du réseau ACLs dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.

Tâches utilisant le mode réseau awsvpc dans un sous-réseau privé

Effectuez la configuration suivante en fonction du runbook :

- Choisissez DISABLED pour attribuer automatiquement une adresse IP publique lors du lancement de la tâche.
- Configurez une passerelle NAT dans votre VPC pour acheminer les demandes vers Internet. Pour plus d'informations, consultez NAT Gateways dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.
- La table de routage du sous-réseau de tâches doit comporter une route pour le trafic vers la passerelle NAT.

  Pour plus d'informations, consultez la section Ajouter et supprimer des itinéraires d'une table de routage dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.

Type de passerelle	Destination de la table de routage	Cible de la table de routage
NAT	0.0.0.0/0	ID de passerelle NAT

- Si le sous-réseau de tâches possède une ACL réseau, les règles ACL suivantes sont requises :
  - Règle sortante qui autorise le trafic sur les ports 1024 à 65535.
  - Règle entrante qui autorise le trafic TCP sur le port 443.

  Pour plus d'informations sur la configuration des règles, consultez la section Contrôler le trafic vers les sous-réseaux à l'aide du réseau ACLs dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.

Tâches n'utilisant pas le mode réseau awsvpc dans un sous-réseau public

Effectuez la configuration suivante en fonction du runbook :

- Choisissez Activer l'attribution automatique de l'adresse IP sous Mise en réseau pour EC2 les instances Amazon lorsque vous créez le cluster.

  Cette option attribue une adresse IP publique à l'interface réseau principale de l'instance.
- Vous avez besoin d'une passerelle pour gérer le trafic Internet. La table de routage du sous-réseau de l'instance doit comporter un itinéraire pour le trafic vers la passerelle.

Pour plus d'informations, consultez la section Ajouter et supprimer des itinéraires d'une table de routage dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.

Type de passerelle	Destination de la table de routage	Cible de la table de routage
NAT	0.0.0.0/0	ID de passerelle NAT
Passerelle Internet	0.0.0.0/0	ID de passerelle Internet

- Si le sous-réseau de l'instance possède une ACL réseau, les règles ACL suivantes sont requises :
  - Règle sortante qui autorise le trafic sur les ports 1024 à 65535.
  - Règle entrante qui autorise le trafic TCP sur le port 443.

  Pour plus d'informations sur la configuration des règles, consultez la section Contrôler le trafic vers les sous-réseaux à l'aide du réseau ACLs dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.

Tâches n'utilisant pas le mode réseau awsvpc dans un sous-réseau privé

Effectuez la configuration suivante en fonction du runbook :

- Choisissez Désactiver pour attribuer automatiquement une adresse IP sous Mise en réseau pour EC2 les instances Amazon lorsque vous créez le cluster.
- Configurez une passerelle NAT dans votre VPC pour acheminer les demandes vers Internet. Pour plus d'informations, veuillez consulter NAT Gateways (Passerelles NAT) dans le Guide de l'utilisateur Amazon VPC.
- La table de routage du sous-réseau de l'instance doit comporter une route pour le trafic vers la passerelle NAT.

  Pour plus d'informations, consultez la section Ajouter et supprimer des itinéraires d'une table de routage dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.

Type de passerelle	Destination de la table de routage	Cible de la table de routage
NAT	0.0.0.0/0	ID de passerelle NAT

- Si le sous-réseau de tâches possède une ACL réseau, les règles ACL suivantes sont requises :
  - Règle sortante qui autorise le trafic sur les ports 1024 à 65535.
  - Règle entrante qui autorise le trafic TCP sur le port 443.

  Pour plus d'informations sur la configuration des règles, consultez la section Contrôler le trafic vers les sous-réseaux à l'aide du réseau ACLs dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.

Afficher les demandes de rôle IAM pour les tâches Amazon ECS

Lorsque vous utilisez un fournisseur pour vos informations d'identification de tâche dans un rôle IAM, les demandes du fournisseur sont enregistrées dans un journal d'audit. Le journal d'audit hérite des mêmes paramètres de rotation du journal que le journal de l'agent de conteneur. Les variables `ECS_LOG_ROLLOVER_TYPE`, `ECS_LOG_MAX_FILE_SIZE_MB` et `ECS_LOG_MAX_ROLL_COUNT` de configuration de l'agent de conteneur peuvent être définies de façon à affecter le comportement du journal d'audit. Pour de plus amples informations, veuillez consulter Paramètres de configuration du journal de l'agent de conteneur Amazon ECS.

Pour l'agent de conteneur version 1.36.0 et ultérieure, le journal d'audit se trouve à l'adresse `/var/log/ecs/audit.log`. Lorsque le journal est tourné, un horodatage au format *YYYY-MM-DD-HH* est ajouté à la fin du nom du fichier journal.

Pour l'agent de conteneur version 1.35.0 et antérieure, le journal d'audit se trouve à l'adresse `/var/log/ecs/audit.log.`*YYYY-MM-DD-HH*.

Le format d'entrée de journal est le suivant :

- Horodatage
- Code de réponse HTTP
- Numéro de port et adresse IP d'origine de la demande
- URI relative du fournisseur d'informations d'identification

- L'agent utilisateur qui a effectué la demande
- L'ARN de la tâche à laquelle appartient le conteneur demandeur
- Le nom d'API `GetCredentials` et son numéro de version
- Le nom du cluster Amazon ECS dans lequel l'instance de conteneur est enregistrée
- L'ARN d'instance de conteneur

Vous pouvez utiliser la commande suivante pour visualiser les fichiers journaux.

```
cat /var/log/ecs/audit.log.2016-07-13-16
```

Sortie :

```
2016-07-13T16:11:53Z 200 172.17.0.5:52444 "/v1/credentials" "python-requests/2.7.0
 CPython/2.7.6 Linux/4.4.14-24.50.amzn1.x86_64" TASK_ARN GetCredentials
 1 CLUSTER_NAME CONTAINER_INSTANCE_ARN
```

Affichage des messages relatifs aux événements du service Amazon ECS

Lorsque vous résolvez un problème lié à un service, vous devez d'abord vérifier les informations de diagnostic dans le journal des événements du service. Vous pouvez consulter les événements de service à l'aide de l'`DescribeServices`API AWS CLI, du, ou en utilisant le AWS Management Console.

Lorsque vous affichez des messages d'événements de service à l'aide de l'API Amazon ECS, seuls les événements du planificateur de service sont renvoyés. Ceux-ci comprennent le placement des tâches les plus récents et les événements d'état de l'instance. Toutefois, la console Amazon ECS affiche les événements de service à partir des sources suivantes.

- Placement des tâches et événements d'état de l'instance à partir du planificateur de service Amazon ECS service. Ces événements ont pour préfixe « service ***(service-name)*** ». Pour que cette vue d'événements soit utile, nous n'affichons que les événements `100` les plus récents. Les messages d'événement dupliqués sont omis tant que la cause n'est pas résolue ou que six heures ne se sont pas écoulées. Si la cause n'est pas résolue dans les six heures, vous recevez un autre message d'événement de service correspondant à cette cause.

- Événements de service Auto Scaling. Ces événements ont le préfixe Message et se produisent uniquement lorsqu'un service est configuré avec une politique de dimensionnement Application Auto Scaling.

Procédez comme suit pour afficher vos messages d'événement de service actuels.

Console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Choisissez le service à inspecter.
5. Pour afficher les messages, choisissez Déploiements et événements, sous Événements.

AWS CLI

Utilisez la commande describe-services pour afficher les messages d'événement de service pour un service spécifié.

L' AWS CLI exemple suivant décrit le *service-name* service du *default* cluster, qui fournira les derniers messages d'événements de service.

```
aws ecs describe-services \
    --cluster default \
    --services service-name \
    --region us-west-2
```

Messages relatifs aux événements du service Amazon ECS

Voici quelques exemples de messages d'événement de service que vous pouvez rencontrer dans la console Amazon ECS.

service (*service-name*) a atteint un état stable.

Le planificateur de service envoie un événement de `service (service-name) has reached a steady state.` service lorsque le service est sain et atteint le nombre de tâches souhaité, atteignant ainsi un état stable.

Le planificateur de service rapporte l'état de façon régulière, vous pouvez donc recevoir ce message plusieurs fois.

service (*service-name*) n'a pas pu placer de tâche car aucune instance de conteneur ne répondait à toutes ses exigences.

Le planificateur de services envoie ce message d'événement lorsqu'il ne trouve pas les ressources disponibles pour ajouter une autre tâche. Les causes possibles sont les suivantes :

Utilisez des fournisseurs de capacité pour dimensionner automatiquement vos EC2 instances. Pour de plus amples informations, veuillez consulter Fournisseurs de capacité Amazon ECS pour le type de EC2 lancement .

Si vous aviez l'intention de faire appel à un fournisseur de capacité, assurez-vous que vous adoptez une stratégie de fournisseur de capacité ou que vous avez une stratégie de fournisseur de capacité par défaut associée à votre cluster et que vous ne transmettez pas le type de lancement et la stratégie du fournisseur de capacité en entrée

Aucune instance de conteneur n'a été trouvée dans votre cluster

Si aucune instance de conteneur n'est enregistrée dans le cluster dans lequel vous tentez d'exécuter une tâche, vous recevez cette erreur. Vous devez ajouter des instances de conteneur à votre cluster. Pour de plus amples informations, veuillez consulter Lancement d'une instance de conteneur Amazon ECS Linux.

Nombre de ports insuffisant

Si votre tâche utilise le mappage de port hôte fixe (si, par exemple, elle emploie le port 80 sur l'hôte pour un serveur web), vous devez avoir au moins une instance de conteneur par tâche, car un seul conteneur peut utiliser un même port hôte à la fois. Vous devez ajouter des instances de conteneur à votre cluster ou réduire votre nombre de tâches souhaitées.

Nombre de ports enregistrés trop important

L'instance de conteneur correspondante la plus proche pour le placement des tâches ne peut pas dépasser la limite de ports réservés maximale autorisée de 100 ports hôtes par instance de conteneur. L'utilisation du mappage de port hôte dynamique peut résoudre le problème.

Port déjà utilisé

La définition de tâche de cette tâche utilise le même port dans son mappage de ports qu'une tâche déjà en cours d'exécution sur l'instance de conteneur choisie. Le message d'événement de service devrait contenir l'ID d'instance de conteneur choisie dans le message ci-dessous.

```
The closest matching container-instance is already using a port required by your
 task.
```

Mémoire insuffisante

Si votre définition de tâche spécifie 1 000 Mio de mémoire et que les instances de conteneur de votre cluster ont chacune 1 024 Mio de mémoire, vous ne pouvez exécuter qu'une seule copie de cette tâche par instance de conteneur. Vous pouvez essayer avec moins de mémoire dans votre définition de tâche afin de pouvoir lancer plusieurs tâches par instance de conteneur, ou lancer plusieurs instances de conteneur dans votre cluster.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire d'instance de conteneur Amazon ECS Linux.

UC insuffisante

Une instance de conteneur comporte 1 024 unités d'UC pour chaque cœur de processeur. Si votre définition de tâche spécifie 1 000 unités d'UC, et que les instances de conteneur de votre cluster ont chacune 1 024 unités d'UC, vous ne pouvez exécuter qu'une seule copie de cette tâche par instance de conteneur. Vous pouvez essayer avec moins d'unités d'UC dans votre définition de tâche pour pouvoir lancer plusieurs tâches par instance de conteneur ou plusieurs instances de conteneur dans votre cluster.

Pas suffisamment de points d'attache ENI disponibles

Les tâches qui utilisent le mode réseau `awsvpc` reçoivent chacune leur propre interface réseau Elastic (ENI), qui est attachée à l'instance de conteneur qui l'héberge. Le nombre d' EC2 instances Amazon ENIs pouvant y être associées est limité et aucune instance de conteneur du cluster ne dispose d'une capacité ENI disponible.

La limite ENI pour les instances de conteneur individuelles varie selon les conditions suivantes :

- Si vous n'avez pas opté pour le paramètre de compte `awsvpcTrunking`, la limite ENI pour chaque instance de conteneur dépend du type de l'instance. Pour plus d'informations, consultez la section Adresses IP par interface réseau et par type d'instance dans le guide de EC2 l'utilisateur Amazon.

- Si vous avez activé le paramétrage du `awsvpcTrunking` compte mais que vous n'avez pas lancé de nouvelles instances de conteneur en utilisant un type d'instance pris en charge après votre inscription, la limite ENI pour chaque instance de conteneur est toujours à la valeur par défaut. Pour plus d'informations, consultez la section Adresses IP par interface réseau et par type d'instance dans le guide de EC2 l'utilisateur Amazon.
- Si vous avez activé les paramètres du `awsvpcTrunking` compte et que vous avez lancé de nouvelles instances de conteneur à l'aide d'un type d'instance pris en charge après votre inscription, d'autres ENIs sont disponibles. Pour de plus amples informations, veuillez consulter Instances prises en charge pour augmenter le nombre d'interfaces réseau de conteneurs Amazon ECS.

Pour en savoir plus sur l'acceptation du paramètre de compte `awsvpcTrunking`, consultez Augmenter les interfaces réseau des instances de conteneur Linux Amazon ECS.

Vous pouvez ajouter des instances de conteneur à votre cluster afin de mettre à disposition davantage de cartes réseau.

Attribut requis manquant dans l'instance de conteneur

Certains paramètres de définition de tâche nécessitent l'installation d'une version spécifique de l'API Docker à distance sur l'instance de conteneur. D'autres, telles que les options du pilote de journalisation, exigent que les instances de conteneur enregistrent ces pilotes de journal avec la variable de configuration d'agent `ECS_AVAILABLE_LOGGING_DRIVERS`. Si votre définition de tâche contient un paramètre qui nécessite un attribut d'instance de conteneur spécifique et que vous ne disposez d'aucune instance de conteneur pouvant satisfaire à cette exigence, la tâche ne peut pas être placée.

Cette erreur est souvent due au fait que votre service utilise des tâches utilisant le mode `awsvpc` réseau et le type de EC2 lancement. Le cluster que vous avez spécifié ne possède aucune instance de conteneur enregistrée dans le même sous-réseau que celui spécifié `awsvpcConfiguration` lors de la création du service.

Vous pouvez utiliser le AWSSupport-TroubleshootECSContainerInstance runbook pour résoudre les problèmes. Le runbook vérifie si les données utilisateur de l'instance contiennent les informations de cluster correctes, si le profil de l'instance contient les autorisations requises et vérifie les problèmes de configuration réseau. Pour plus d'informations, consultez AWSSupport-TroubleshootECSContainerInstance dans le AWS Systems Manager Guide de l'utilisateur de la référence du dossier d'exploitation d'automatisation.

Pour plus d'informations sur les attributs nécessaires pour les paramètres de définition de tâche spécifiques et les variables de configuration d'agent, consultez Paramètres de définition des tâches Amazon ECS pour le type de lancement Fargate et Configuration de l'agent de conteneur Amazon ECS.

service (*service-name*) n'a pas pu placer de tâche car aucune instance de conteneur ne répondait à toutes ses exigences. L'instance de conteneur correspondante la plus proche ne *container-instance-id* dispose pas d'un nombre suffisant d'unités CPU disponibles.

L'instance de conteneur correspondante la plus proche pour le placement des tâches ne contient pas suffisamment d'unités de processeur pour répondre aux exigences de la définition de la tâche. Vérifiez les exigences en matière d'UC dans les paramètres de définition de conteneur et de taille de tâche de la définition de tâche.

service (*service-name*) n'a pas pu placer de tâche car aucune instance de conteneur ne répondait à toutes ses exigences. L'instance de conteneur correspondante la plus proche *container-instance-id* a rencontré l'erreur « AGENT ».

L'agent de conteneur Amazon ECS de l'instance de conteneur correspondante la plus proche pour le placement des tâches est déconnecté. Si vous pouvez vous connecter à l'instance de conteneur avec SSH, vous pouvez examiner les journaux d'agent. Pour plus d'informations, consultez Paramètres de configuration du journal de l'agent de conteneur Amazon ECS. Vous devez également vérifier que l'agent est en cours d'exécution sur l'instance. Si vous utilisez l'AMI optimisée pour Amazon ECS, vous pouvez essayer d'arrêter et de redémarrer l'agent avec la commande suivante.

- Pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS et l'AMI Amazon Linux 2023 optimisée pour Amazon ECS

```
sudo systemctl restart ecs
```

- Pour l'AMI Amazon Linux optimisée pour Amazon ECS

```
sudo stop ecs && sudo start ecs
```

le service (*service-name*) (tâche*task-id*) (instance*instance-id*) est défectueux dans (elb*elb-name*) en raison de (raison pour laquelle l'instance a échoué à au moins un UnhealthyThreshold certain nombre de contrôles de santé consécutifs.)

Ce service est enregistré avec un équilibreur de charge et les surveillances de l'état de cet équilibreur échouent. Le message inclut l'ID de tâche pour aider à identifier la tâche spécifique pour laquelle les tests de santé échouent. Pour de plus amples informations, veuillez consulter Résolution des problèmes liés aux équilibreurs de charge de service dans Amazon ECS.

service (*service-name*) ne parvient pas à démarrer régulièrement les tâches avec succès.

Ce service contient des tâches qui n'ont pas pu démarrer après plusieurs tentatives consécutives. À ce stade, le planificateur de service commence à augmenter progressivement le délai entre les tentatives. Vous devez déterminer pourquoi le lancement de vos tâches échoue. Pour de plus amples informations, veuillez consulter Logique de régulation du service Amazon ECS.

Une fois le service mis à jour (mise à jour de la définition de tâche, par exemple), le planificateur de service se comporte de nouveau normalement.

les opérations de service (*service-name*) sont limitées. Réessayez ultérieurement.

Ce service ne peut pas lancer plus de tâches à cause des limitations de l'API. Lorsque le planificateur de service peut lancer plus de tâches, il reprend.

Pour demander une augmentation de quota de limite de débit d'API, ouvrez la page du Centre AWS Support, connectez-vous si nécessaire et choisissez Create case (Créer une demande). Sélectionnez Service Limit increase (Augmentation des limites de service). Remplissez et envoyez le formulaire.

service (*service-name*) n'a pas pu arrêter ou démarrer des tâches pendant un déploiement en raison de la configuration du déploiement du service. Mettez à jour la valeur minimumHealthyPercent ou MaximumPercent et réessayez.

Ce service ne peut pas arrêter ou démarrer des tâches pendant un déploiement de service en raison de la configuration du déploiement. La configuration de déploiement comprend les `maximumPercent` valeurs `minimumHealthyPercent` et, qui sont définies lors de la création du service. Ces valeurs peuvent également être mises à jour sur un service existant.

Le `minimumHealthyPercent` représente la limite inférieure du nombre de tâches qui doivent être exécutées pour un service lors d'un déploiement ou lorsqu'une instance de conteneur est épuisée.

Il s'agit d'un pourcentage du nombre de tâches souhaité pour le service. Cette valeur est arrondie à la valeur supérieure. Par exemple, si le pourcentage de santé minimum est de quatre `50` et que le nombre de tâches souhaité est de quatre, le planificateur peut arrêter deux tâches existantes avant d'en démarrer deux nouvelles. De même, si le pourcentage de santé minimum est de 75 % et que le nombre de tâches souhaité est de deux, le planificateur ne peut pas arrêter de tâche car la valeur résultante est également de deux.

Le `maximumPercent` représente la limite supérieure du nombre de tâches qui doivent être exécutées pour un service lors d'un déploiement ou lorsqu'une instance de conteneur est épuisée. Il s'agit d'un pourcentage du nombre de tâches souhaité pour un service. Cette valeur est arrondie à la valeur inférieure. Par exemple, si le pourcentage maximal est de quatre `200` et que le nombre de tâches souhaité est de quatre, le planificateur peut démarrer quatre nouvelles tâches avant d'arrêter quatre tâches existantes. De même, si le pourcentage maximal est `125` et que le nombre de tâches souhaité est de trois, le planificateur ne peut pas démarrer de tâche car la valeur résultante est également de trois.

Lorsque vous définissez un pourcentage d'état minimum ou un pourcentage maximal, vous devez vous assurer que le planificateur peut arrêter ou démarrer au moins une tâche lorsqu'un déploiement est déclenché.

service (***`service-name`***) n'a pas pu placer de tâche. Motif : Vous avez atteint la limite du nombre de tâches que vous pouvez exécuter simultanément

Vous pouvez demander une augmentation de quota pour la ressource qui a provoqué l'erreur. Pour de plus amples informations, veuillez consulter Service Quotas. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

service (***`service-name`***) n'a pas pu placer de tâche. Motif : Erreur interne.

La raison possible de cette erreur est la suivante :

Le service ne peut pas démarrer une tâche car un sous-réseau se trouve dans une zone de disponibilité non prise en charge.

Pour plus d'informations sur les Régions Fargate et les zones de disponibilités prises en charge, consultez the section called “AWS Régions de Fargate”.

Pour plus d'informations sur la façon d'afficher la zone de disponibilité de sous-réseau, consultez Afficher votre sous-réseau dans le Guide de l'utilisateur Amazon VPC.

service (*service-name*) n'a pas pu placer de tâche. Motif : la configuration CPU demandée est supérieure à votre limite.

Vous pouvez demander une augmentation de quota pour la ressource qui a provoqué l'erreur. Pour de plus amples informations, veuillez consulter Service Quotas. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

service (*service-name*) n'a pas pu placer de tâche. Motif : la configuration MEMORY demandée est supérieure à votre limite.

Vous pouvez demander une augmentation de quota pour la ressource qui a provoqué l'erreur. Pour de plus amples informations, veuillez consulter Service Quotas. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

service (*service-name*) n'a pas pu placer de tâche. Raison : vous avez atteint la limite du nombre de v CPUs que vous pouvez exécuter simultanément

AWS Fargate passe des quotas basés sur le nombre de tâches aux quotas basés sur le vCPU.

Vous pouvez demander une augmentation de quota pour le quota basé sur le VCPU Fargate. Pour de plus amples informations, veuillez consulter Service Quotas. Pour demander une augmentation de quota Fargate, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

service (*service-name*) n'a pas pu atteindre l'état d'équilibre car task set (*taskSet-ID*) n'a pas pu évoluer. Raison : le nombre de tâches protégées est supérieur au nombre de tâches souhaité

Le service comporte plus de tâches protégées que le nombre de tâches souhaité. Vous pouvez effectuer l'une des actions suivantes :

- Attendez que la protection des tâches en cours expire pour pouvoir y mettre fin.
- Déterminez quelles tâches peuvent être arrêtées et utilisez l'`UpdateTaskProtection`API avec l'`protectionEnabled`option définie sur `false` pour désactiver la protection de ces tâches.
- Augmentez le nombre de tâches souhaité pour le service à un nombre supérieur au nombre de tâches protégées.

service (***service-name***) n'a pas pu atteindre l'état d'équilibre. Motif : aucune instance de conteneur n'a été trouvée dans votre fournisseur de capacité.

Le planificateur de services envoie ce message d'événement lorsqu'il ne trouve pas les ressources disponibles pour ajouter une autre tâche. Les causes possibles sont les suivantes :

Aucun fournisseur de capacité n'est associé au cluster

`describe-services`À utiliser pour vérifier qu'un fournisseur de capacité est associé au cluster. Vous pouvez mettre à jour la stratégie du fournisseur de capacité pour le service.

Vérifiez que la capacité du fournisseur de capacité est disponible. Dans le cas du type de EC2 lancement, assurez-vous que les instances de conteneur répondent aux exigences de définition des tâches.

Aucune instance de conteneur n'a été trouvée dans votre cluster

Si aucune instance de conteneur n'est enregistrée dans le cluster dans lequel vous tentez d'exécuter une tâche, vous recevez cette erreur. Vous devez ajouter des instances de conteneur à votre cluster. Pour de plus amples informations, veuillez consulter Lancement d'une instance de conteneur Amazon ECS Linux.

Nombre de ports insuffisant

Si votre tâche utilise un mappage de port hôte fixe (par exemple, votre tâche utilise le port 80 sur l'hôte pour un serveur Web), vous devez disposer d'au moins une instance de conteneur par tâche. Un seul conteneur peut utiliser un seul port hôte à la fois. Vous devez ajouter des instances de conteneur à votre cluster ou réduire votre nombre de tâches souhaitées.

Nombre de ports enregistrés trop important

L'instance de conteneur correspondante la plus proche pour le placement des tâches ne peut pas dépasser la limite de ports réservés maximale autorisée de 100 ports hôtes par instance de conteneur. L'utilisation du mappage de port hôte dynamique peut résoudre le problème.

Port déjà utilisé

La définition de tâche de cette tâche utilise le même port dans son mappage de ports qu'une tâche déjà en cours d'exécution sur l'instance de conteneur choisie. Le message d'événement de service devrait contenir l'ID d'instance de conteneur choisie dans le message ci-dessous.

```
The closest matching container-instance is already using a port required by your
 task.
```

Mémoire insuffisante

Si votre définition de tâche spécifie 1 000 Mio de mémoire et que les instances de conteneur de votre cluster ont chacune 1 024 Mio de mémoire, vous ne pouvez exécuter qu'une seule copie de cette tâche par instance de conteneur. Vous pouvez essayer avec moins de mémoire dans votre définition de tâche afin de pouvoir lancer plusieurs tâches par instance de conteneur, ou lancer plusieurs instances de conteneur dans votre cluster.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire d'instance de conteneur Amazon ECS Linux.

Pas suffisamment de points d'attache ENI disponibles

Les tâches qui utilisent le mode réseau `awsvpc` reçoivent chacune leur propre interface réseau Elastic (ENI), qui est attachée à l'instance de conteneur qui l'héberge. Le nombre d' EC2 instances Amazon ENIs pouvant y être associées est limité, et aucune instance de conteneur du cluster ne dispose d'une capacité ENI disponible.

La limite ENI pour les instances de conteneur individuelles varie selon les conditions suivantes :

- Si vous n'avez pas opté pour le paramètre de compte `awsvpcTrunking`, la limite ENI pour chaque instance de conteneur dépend du type de l'instance. Pour plus d'informations, consultez la section Adresses IP par interface réseau et par type d'instance dans le guide de EC2 l'utilisateur Amazon.
- Si vous avez activé le paramétrage du `awsvpcTrunking` compte mais que vous n'avez pas lancé de nouvelles instances de conteneur en utilisant un type d'instance pris en charge après votre inscription, la limite ENI pour chaque instance de conteneur est toujours à la valeur par défaut. Pour plus d'informations, consultez la section Adresses IP par interface réseau et par type d'instance dans le guide de EC2 l'utilisateur Amazon.
- Si vous avez activé les paramètres du `awsvpcTrunking` compte et que vous avez lancé de nouvelles instances de conteneur à l'aide d'un type d'instance pris en charge après votre inscription, d'autres ENIs sont disponibles. Pour de plus amples informations, veuillez consulter Instances prises en charge pour augmenter le nombre d'interfaces réseau de conteneurs Amazon ECS.

Pour en savoir plus sur l'acceptation du paramètre de compte `awsvpcTrunking`, consultez Augmenter les interfaces réseau des instances de conteneur Linux Amazon ECS.

Vous pouvez ajouter des instances de conteneur à votre cluster afin de mettre à disposition davantage de cartes réseau.

Attribut requis manquant dans l'instance de conteneur

Certains paramètres de définition de tâche nécessitent l'installation d'une version spécifique de l'API Docker à distance sur l'instance de conteneur. D'autres, telles que les options du pilote de journalisation, exigent que les instances de conteneur enregistrent ces pilotes de journal avec la variable de configuration d'agent `ECS_AVAILABLE_LOGGING_DRIVERS`. Si votre définition de tâche contient un paramètre qui nécessite un attribut d'instance de conteneur spécifique et que vous ne disposez d'aucune instance de conteneur pouvant satisfaire à cette exigence, la tâche ne peut pas être placée.

Cette erreur est souvent due au fait que votre service utilise des tâches utilisant le mode `awsvpc` réseau et le type de EC2 lancement et que le cluster que vous avez spécifié ne possède aucune instance de conteneur enregistrée dans le même sous-réseau que celui spécifié `awsvpcConfiguration` lors de la création du service.

Vous pouvez utiliser le AWSSupport-TroubleshootECSContainerInstance runbook pour résoudre les problèmes. Le runbook vérifie si les données utilisateur de l'instance contiennent les informations de cluster correctes, si le profil de l'instance contient les autorisations requises et vérifie les problèmes de configuration réseau. Pour plus d'informations, consultez AWSSupport-TroubleshootECSContainerInstance dans le AWS Systems Manager Guide de l'utilisateur de la référence du dossier d'exploitation d'automatisation.

Pour plus d'informations sur les attributs nécessaires pour les paramètres de définition de tâche spécifiques et les variables de configuration d'agent, consultez Paramètres de définition des tâches Amazon ECS pour le type de lancement Fargate et Configuration de l'agent de conteneur Amazon ECS.

service (***`service-name`***) n'a pas pu placer de tâche. Motif : la capacité n'est pas disponible pour le moment. Veuillez réessayer ultérieurement ou dans une autre zone de disponibilité.

Il n'y a actuellement aucune capacité disponible pour exécuter votre service.

Vous pouvez effectuer l'une des actions suivantes :

- Attendez que la capacité de Fargate EC2 ou les instances de conteneur soient disponibles.
- Relancez le service et spécifiez des sous-réseaux supplémentaires.

échec du déploiement de service (***service-name***) : les tâches n'ont pas pu démarrer.

Les tâches de votre service n'ont pas pu démarrer.

Pour plus d'informations sur la façon de déboguer les tâches arrêtées, consultez. Messages d'erreur relatifs aux tâches interrompues par Amazon ECS

service (***service-name***) Le délai d'attente du démarrage de l'agent Amazon ECS a expiré. Veuillez consulter les journaux à l'adresse «/var/log/ecs/ecs-agent.log ».

L'agent de conteneur Amazon ECS de l'instance de conteneur correspondante la plus proche pour le placement des tâches est déconnecté. Si vous pouvez vous connecter à l'instance de conteneur via SSH, vous pouvez examiner les journaux de l'agent. Pour de plus amples informations, veuillez consulter Paramètres de configuration du journal de l'agent de conteneur Amazon ECS. Vous devez également vérifier que l'agent est en cours d'exécution sur l'instance. Si vous utilisez l'AMI optimisée pour Amazon ECS, vous pouvez essayer d'arrêter et de redémarrer l'agent avec la commande suivante.

- Pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS

```
sudo systemctl restart ecs
```

- Pour l'AMI Amazon Linux optimisée pour Amazon ECS

```
sudo stop ecs && sudo start ecs
```

service (***service-name***) task set (***taskSet-ID***) (task***task-id***) n'est pas sain dans le groupe cible (***targetGroup-ARN)***) en raison de. **TARGET GROUP IS NOT FOUND**

La tâche définie pour le service échoue aux tests de santé car le groupe cible est introuvable. Le message inclut l'ID de tâche pour aider à identifier la tâche spécifique pour laquelle les tests de santé échouent. Vous devez supprimer et recréer le service. Ne supprimez aucun groupe cible Elastic Load Balancing à moins que le service Amazon ECS correspondant ne soit déjà supprimé.

service (***service-name***) task set (***taskSet-ID***) (task***task-id***) n'est pas sain dans le groupe cible (***targetGroup-ARN)***) en raison de. **TARGET IS NOT FOUND**

La tâche définie pour le service échoue aux tests de santé car la cible est introuvable. Le message inclut l'ID de tâche pour aider à identifier la tâche spécifique pour laquelle les tests de santé échouent.

Messages d'événements non conformes au service Amazon ECS

Vous trouverez ci-dessous des exemples de messages indiquant un dysfonctionnement du service.

EC2: (service***service-name***) (tâche***task-id***) (instance***instance-id***) (port***port-number***) ne fonctionne pas correctement dans (groupe cible***target-group-name***) en raison de (raison) ***failure-reason***

Ce message indique qu'une tâche exécutée sur une EC2 instance échoue aux tests de santé. Pour plus d'informations, consultez les ressources suivantes :

- Comment puis-je faire en sorte que mes tâches Amazon ECS utilisant le type de EC2 lancement Amazon passent avec succès le test de santé d'Application Load Balancer ?

EC2 avec TaskSet : (service***service-name***, TaskSet***taskSet-id***) (tâche***task-id***) (instance***instance-id***) (port***port-number***) est défectueux dans (groupe cible) en raison de (raison***target-group-name***) ***failure-reason***

Ce message indique qu'une tâche d'un ensemble de tâches exécutée sur une EC2 instance échoue aux tests de santé. Pour plus d'informations, consultez les ressources suivantes :

- Comment puis-je faire en sorte que mes tâches Amazon ECS utilisant le type de EC2 lancement Amazon passent avec succès le test de santé d'Application Load Balancer ?

Fargate : (***service-name***service) (***task-id***tâche) (***port-number***port) ne fonctionne pas correctement dans (***target-group-name***groupe cible) en raison de (raison) ***failure-reason***

Ce message indique qu'une tâche Fargate échoue aux tests de santé.

Pour plus d'informations sur la résolution des échecs de contrôle de santé dans les tâches Fargate, consultez Comment résoudre les échecs des tests de santé pour les tâches Amazon ECS sur Fargate ? .

Fargate avec TaskSet : (*service-name*service, *taskSet-id* TaskSet) (*task-id*tâche) (*port-number*port) ne fonctionne pas correctement dans (groupe *target-group-name* cible) en raison de (raison) *failure-reason*

Ce message indique qu'une tâche d'un ensemble de tâches exécuté sur Fargate échoue aux tests de santé.

Pour plus d'informations sur la résolution des échecs de contrôle de santé dans les tâches Fargate, consultez Comment résoudre les échecs des tests de santé pour les tâches Amazon ECS sur Fargate ? .

Messages relatifs aux événements du service de rééquilibrage du service Amazon ECS Availability Zone

Vous trouverez ci-dessous des exemples de messages d'événements de service susceptibles de s'afficher.

service (*service-name*) n'est pas équilibré entre les *number-tasks* tâches *number-tasks* dans *Availability Zone 1Availability Zone 2*, dans et *number-tasks* dans*Availability Zone 3*. Rééquilibrage AZ en cours.

Le planificateur de services envoie un événement de `service (`*`service-name`*`) is not AZ balanced` service lorsque le nombre de tâches n'est pas réparti uniformément entre les zones de disponibilité. Il n'y a aucune mesure à prendre. Il s'agit d'un événement d'information.

service (*service-name*) est équilibré en AZ avec *number-tasks* les tâches dans*Availability Zone 1*, *number-tasks* les tâches dans *Availability Zone 2* et *number-tasks* les tâches dans*Availability Zone 3*.

Le planificateur de services envoie un événement de `service (`*`service-name`*`) is AZ balanced` service lorsque le rééquilibrage des services de zone de disponibilité est terminé. Il n'y a aucune mesure à prendre. Il s'agit d'un événement d'information.

service-namea commencé ***number-tasks*** des tâches dans ***Availability Zone*** AZ Rebalance :***task-ids***.

Le planificateur de services envoie un événement*service-name*/*task-set-name*a démarré *number* des tâches dans le *Availability Zone* service lorsqu'il démarre des tâches dans une zone de disponibilité en raison du rééquilibrage des services. Il n'y a aucune mesure à prendre. Il s'agit d'un événement d'information.

service-namea cessé ***number-tasks*** d'exécuter des tâches en ***Availability Zone*** raison du rééquilibrage AZ :***task-id***.

Le planificateur de services envoie un événement*service-name*/*task-set-name*a arrêté les *number* tâches dans le *Availability Zone* service lorsqu'il arrête des tâches dans une zone de disponibilité en raison du rééquilibrage des services. Il n'y a aucune mesure à prendre. Il s'agit d'un événement d'information.

service (***service-name***) n'est pas en mesure de placer une tâche ***Availability Zone*** car aucune instance de conteneur ne répond à toutes ses exigences.

Le planificateur de services envoie un *service-name* événement Impossible de placer une tâche dans le *Availability Zone* service car aucune instance de conteneur ne répond à toutes ses exigences. Pour résoudre ce problème, lancez des instances dans la zone de disponibilité.

service (***service-name***) ne parvient pas à placer une tâche dedans***Availability Zone***.

Le planificateur de services envoie un *service-name* événement « Impossible de placer une tâche dans le *Availability Zone* service » lorsque vous utilisez le type de lancement Fargate et qu'aucune capacité n'est disponible.

Vous pouvez ajouter des sous-réseaux supplémentaires dans la zone de disponibilité dans le message d'erreur, ou contacter Support pour obtenir de la capacité supplémentaire.

service (***service-name***) n'a pas pu effectuer AZ Rebalance car il n'***task-set-name***a pas pu être intégré en raison de***reason***.

Le planificateur de services envoie un *service-name* message « Impossible » à AZ Rebalance car celui-ci n'*task-set-name*a pas pu être redimensionné en raison d'un événement de *reason* service lorsque vous utilisez la protection intégrée des tâches.

Vous pouvez effectuer l'une des actions suivantes :

- Attendez que la protection des tâches en cours expire pour pouvoir y mettre fin.
- Déterminez quelles tâches peuvent être arrêtées et utilisez l'`UpdateTaskProtection`API avec l'`protectionEnabled`option définie sur `false` pour désactiver la protection de ces tâches.
- Augmentez le nombre de tâches souhaité pour le service à un nombre supérieur au nombre de tâches protégées.

service (*`service-name`*) a arrêté AZ Rebalancing.

Le planificateur de services envoie un événement d'*`service-name`*arrêt du service AZ Rebalancing lorsque l'opération de rééquilibrage de la zone de disponibilité s'arrête. Il s'agit d'un événement d'information. Amazon ECS envoie des événements supplémentaires qui fournissent des informations supplémentaires.

Résolution des problèmes liés aux équilibreurs de charge de service dans Amazon ECS

Les services Amazon ECS service peuvent enregistrer des tâches auprès d'un équilibreur de charge Elastic Load Balancing. Les erreurs de configuration d'équilibreur de charge sont des causes courantes de l'arrêt des tâches. Si vos tâches arrêtées ont été lancées par des services qui utilisent un équilibreur de charge, pensez aux causes possibles suivantes.

Le rôle lié au service Amazon ECS n'existe pas

Le rôle lié à un service Amazon ECS service permet aux services Amazon ECS service d'enregistrer des instances de conteneur avec des équilibreurs de charge Elastic Load Balancing. Le rôle lié à un service doit être créé dans votre compte. Pour de plus amples informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS.

Groupe de sécurité des instances de conteneurs

Si votre conteneur est mappé au port 80 sur votre instance de conteneur, votre groupe de sécurité d'instance de conteneur doit autoriser le trafic entrant sur le port 80 pour que les surveillances de l'état de l'équilibreur de charge réussissent.

L'équilibreur de charge Elastic Load Balancing n'est pas configuré pour toutes les zones de disponibilité

Votre équilibreur de charge doit être configuré pour utiliser toutes les zones de disponibilité d'une région, ou au moins toutes les zones de disponibilité dans lesquelles se trouvent vos instances de conteneur. Si un service utilise un équilibreur de charge et lance une tâche sur une instance de conteneur située dans une zone de disponibilité que l'équilibreur de charge n'est pas configuré pour utiliser, la tâche ne passe jamais le test de santé. Cela entraîne l'arrêt de la tâche.

Le bilan de santé de l'équilibreur de charge Elastic Load Balancing est mal configuré

Les paramètres de surveillance de l'état de l'équilibreur de charge peuvent être trop restrictifs ou pointer vers des ressources qui n'existent pas. S'il est déterminé qu'une instance de conteneur est défectueuse, elle est supprimée de l'équilibreur de charge. Veillez à vérifier si les paramètres suivants sont correctement configurés pour votre équilibreur de charge de service.

Ping Port

La valeur Ping Port pour une surveillance de l'état d'équilibreur de charge correspond au port des instances de conteneur vérifiées par l'équilibreur de charge pour en déterminer l'état. Si ce port n'est pas correctement configuré, l'équilibreur de charge est susceptible d'annuler l'enregistrement de votre instance de conteneur. Ce port doit être configuré de manière à utiliser la valeur `hostPort` correspondant au conteneur dans la définition de tâche de votre service que vous utilisez avec la surveillance de l'état.

Ping Path

Cela fait partie du bilan de santé de l'équilibreur de charge. Il s'agit d'un point de terminaison de votre application qui peut renvoyer un code d'état valide (par exemple, 200) lorsque l'application est saine. Cette valeur est souvent définie sur `index.html`, mais si votre service ne répond pas à cette demande, la surveillance de l'état échoue. Si votre conteneur n'a pas de fichier `index.html`, vous pouvez définir la valeur `/` afin de cibler l'URL de base pour l'instance de conteneur.

Response Timeout

Cette valeur correspond au délai dont dispose votre conteneur pour renvoyer une réponse à la commande ping de surveillance de l'état. Si cette valeur est inférieure à la durée nécessaire pour une réponse, la surveillance de l'état échoue.

Health Check Interval

Il s'agit de la durée entre les pings de surveillance de l'état. Plus vos intervalles de surveillance de l'état sont courts, plus votre instance de conteneur peut atteindre rapidement le seuil de défectuosité.

Unhealthy Threshold

Il s'agit du nombre d'échecs possibles des surveillances de l'état avant que votre instance de conteneur ne soit considérée comme étant défectueuse. Si vous avez un seuil d'insalubrité de 2 et un intervalle de 30 secondes entre les bilans de santé, votre tâche dispose de 60 secondes pour répondre au ping du bilan de santé avant qu'elle ne soit considérée comme non saine. Vous pouvez augmenter le seuil de défectuosité ou l'intervalle de surveillance de l'état afin de donner à vos tâches plus de temps pour répondre.

Impossible de mettre à jour le service ***servicename*** : le nom ou le port du conteneur de l'équilibreur de charge ont été modifiés dans la définition de la tâche

Si votre service utilise un équilibreur de charge, vous pouvez utiliser le SDK AWS CLI ou le SDK pour modifier la configuration de l'équilibreur de charge. Pour plus d'informations sur la façon de modifier la configuration, consultez le UpdateServicemanuel Amazon Elastic Container Service API Reference. Si vous mettez à jour la définition de tâche pour le service, le nom et le port de conteneur spécifiés dans la configuration de l'équilibreur de charge doivent rester dans la définition de tâche.

Vous avez atteint la limite du nombre de tâches que vous pouvez exécuter simultanément.

Pour un nouveau compte, vos quotas peuvent être inférieurs aux Service Quotas. Le quota de service de votre compte peut être affiché dans la console Service Quotas. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

Résolution des problèmes liés au dimensionnement automatique du service dans Amazon ECS

Application Auto Scaling désactive les processus d'évolutivité lorsque les déploiements Amazon ECS sont en cours, et ils reprennent une fois le déploiement terminé. Toutefois, pendant les déploiements, les processus de montée en puissance se poursuivent, sauf s'ils sont suspendus. Pour de plus amples informations, veuillez consulter Suspension et reprise de la mise à l'échelle pour Application Auto Scaling.

Résoudre les erreurs de processeur ou de mémoire non valides liées à la définition des tâches Amazon ECS

Lorsque vous enregistrez une définition de tâche à l'aide de l'API Amazon ECS ou AWS CLI, si vous spécifiez une `memory` valeur `cpu` ou une valeur non valide, l'erreur suivante est renvoyée.

```
An error occurred (ClientException) when calling the RegisterTaskDefinition operation:
 Invalid 'cpu' setting for task.
```

> **Note**
>
> Lorsque vous utilisez Terraform, l'erreur suivante peut être renvoyée.
>
> ```
> Error: ClientException: No Fargate configuration exists for given values.
> ```

Pour résoudre ce problème, vous devez spécifier une valeur prise en charge pour l'UC et la mémoire de la tâche dans votre définition de tâche. La `cpu` valeur peut être exprimée en unités CPU ou CPUs en v dans une définition de tâche. Il est converti en un entier indiquant les unités du processeur lorsque la définition de tâche est enregistrée. La `memory` valeur peut être exprimée en MiB ou en Go dans une définition de tâche. Il est converti en un entier indiquant le MiB lorsque la définition de tâche est enregistrée.

Pour les définitions de tâches qui spécifient `FARGATE` le `requiresCompatibilities` paramètre (même s'il `EC2` est également spécifié), vous devez utiliser l'une des valeurs du tableau suivant. Ces valeurs déterminent votre plage de valeurs prises en charge pour le paramètre CPU et mémoire.

Pour les tâches hébergées sur Fargate, le tableau suivant indique les combinaisons de processeur et de mémoire valides. Les valeurs de mémoire du fichier JSON sont spécifiées en Mio. Vous pouvez convertir la valeur en Go en Mio en la multipliant par 1 024. Par exemple, 1 Go = 1 024 Mio.

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
256 (0,25 vCPU)	512 Mio, 1 Go, 2 Go	Linux
512 (0,5 vCPU)	1 Go, 2 Go, 3 Go, 4 Go	Linux

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
1 024 (1 vCPU)	2 Go, 3 Go, 4 Go, 5 Go, 6 Go, 7 Go, 8 Go	Linux, Windows
2 048 (2 vCPU)	Entre 4 Go et 16 Go par incréments de 1 Go	Linux, Windows
4 096 (4 vCPU)	Entre 8 Go et 30 Go par incréments de 1 Go	Linux, Windows
8192 (8 vCPU) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 16 Go et 60 Go par incréments de 4 Go	Linux
16384 (16vCPU) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 32 Go et 120 Go par incréments de 8 Go	Linux

Pour les tâches hébergées sur Amazon EC2, les valeurs du processeur des tâches prises en charge sont comprises entre 0,25 v CPUs et 192 v. CPUs

Le mécanisme de contrôle du processeur diffère entre les types de EC2 lancement Amazon et Fargate :

- Pour les tâches hébergées sur Amazon EC2 : Amazon ECS utilise la durée du processeur et le quota de processeur pour contrôler la taille des tâches, les limites strictes du processeur. Lorsque

vous spécifiez le vCPU dans votre définition de tâche, Amazon ECS traduit la valeur en durée du processeur et en paramètres de quota de processeur qui s'appliquent au. `cgroup`

- Pour les tâches hébergées sur Fargate : Amazon ECS utilise des partages de processeur pour contrôler l'allocation du processeur. Les valeurs de quota et de période du processeur ne sont pas utilisées pour limiter le processeur dans les tâches Fargate.

Pour les EC2 tâches Amazon, le quota de processeur contrôle le temps processeur accordé à un `cgroup` pendant une période de processeur donnée. Les deux paramètres sont exprimés en microsecondes. Lorsque le quota de processeur est égal à la période du processeur, cela signifie que a `cgroup` peut exécuter jusqu'à 100 % sur un vCPU (ou toute autre fraction totalisant 100 % pour plusieurs v). CPUs Le quota du processeur est au maximum de 1 000 000 us et la période du processeur est d'au moins 1 ms. Vous pouvez utiliser ces valeurs pour définir les limites de votre nombre de processeurs. Lorsque vous modifiez la période du processeur sans modifier le quota du processeur, vous disposez de limites effectives différentes de celles que vous avez spécifiées dans votre définition de tâche.

La période de 100 ms permet un v CPUs compris entre 0,125 et 10.

> **Note**
>
> Les paramètres d'UC et de mémoire de niveau tâche sont ignorés pour les conteneurs Windows.

Afficher les journaux des agents de conteneurs Amazon ECS

Amazon ECS stocke les journaux dans le dossier `/var/log/ecs` de vos instances de conteneur. Des journaux sont disponibles à partir de l'agent de conteneur Amazon ECS et du service `ecs-init` qui contrôle l'état de l'agent (démarrage/arrêt) sur l'instance de conteneur. Vous pouvez afficher ces fichiers journaux en vous connectant à une instance de conteneur à l'aide de SSH.

> **Note**
>
> Si vous n'êtes pas sûr de savoir comment collecter tous les journaux de vos instances de conteneur, vous pouvez utiliser le collecteur de journaux d'Amazon ECS. Pour de plus amples informations, veuillez consulter Collecte des journaux de conteneurs avec le collecteur de journaux Amazon ECS .

Système d'exploitation Linux

Le processus `ecs-init` stocke les journaux à l'emplacement `/var/log/ecs/ecs-init.log`.

Le `ecs-init.log` fichier contient des informations sur la gestion du cycle de vie, la configuration et le démarrage de l'agent conteneur.

Vous pouvez utiliser la commande suivante pour visualiser les fichiers journaux.

```
cat /var/log/ecs/ecs-init.log
```

Sortie :

```
2018-02-16T18:13:54Z [INFO] pre-start
2018-02-16T18:13:56Z [INFO] start
2018-02-16T18:13:56Z [INFO] No existing agent container to remove.
2018-02-16T18:13:56Z [INFO] Starting Amazon Elastic Container Service Agent
```

Système d'exploitation Windows

Vous pouvez utiliser le collecteur de journaux Amazon ECS pour Windows. Pour plus d'informations, consultez Amazon ECS Logs Collector pour Windows sur Github.

1. Connectez-vous à votre instance.
2. Ouvrez PowerShell puis exécutez les commandes suivantes avec des privilèges administratifs. Les commandes téléchargent le script et collectent les journaux.

   ```
   Invoke-WebRequest -OutFile ecs-logs-collector.ps1 https://
   raw.githubusercontent.com/awslabs/aws-ecs-logs-collector-for-windows/master/ecs-
   logs-collector.ps1
   .\ecs-logs-collector.ps1
   ```

Vous pouvez activer la journalisation du débogage pour l'agent Amazon ECS et le daemon Docker. Cette option permet au script de collecter les journaux avant d'activer le mode de débogage. Le script redémarre le démon Docker et l'agent Amazon ECS, puis met fin à tous les conteneurs exécutés sur l'instance. Avant d'exécuter la commande suivante, videz l'instance de conteneur et déplacez les tâches importantes vers d'autres instances de conteneur.

Exécutez la commande suivante pour activer la journalisation.

```
.\ecs-logs-collector.ps1 -RunMode debug
```

Collecte des journaux de conteneurs avec le collecteur de journaux Amazon ECS

Si vous n'êtes pas sûr de savoir comment de collecter les différents journaux de vos instances de conteneur, vous pouvez utiliser le collecteur de journaux d'Amazon ECS. Il est disponible GitHub pour Linux et Windows. Le script collecte les journaux généraux du système d'exploitation ainsi que les journaux des agents de conteneur Docker et Amazon ECS, ce qui peut être utile pour les AWS Support cas de dépannage. Puis, il compresse et archive les informations collectées dans un seul fichier qui peut être facilement partagé à des fins de diagnostic. Il prend également en charge l'activation du mode de débogage pour le démon Docker et l'agent de conteneur Amazon ECS sur les variantes d'Amazon Linux (AMI optimisée pour Amazon ECS, par exemple). Actuellement, le collecteur de journaux Amazon ECS prend en charge les systèmes d'exploitation suivants :

- Amazon Linux
- Red Hat Enterprise Linux 7
- Debian 8
- Ubuntu 14.04
- Ubuntu 16.04
- Ubuntu 18.04
- Windows Server 2016

> **Note**
>
> Le code source du collecteur de logs Amazon ECS est disponible GitHub pour Linux et Windows. Nous vous conseillons d'envoyer des requêtes d'extraction pour les modifications que vous souhaitez inclure. Toutefois, Amazon Web Services ne prend actuellement pas en charge l'exécution de copies modifiées de ce logiciel.

Pour télécharger et exécuter le collecteur de journaux Amazon ECS pour Linux

1. Connectez-vous à votre instance de conteneur.

2. Téléchargez le script de collecteur de journaux Amazon ECS.

```
curl -O https://raw.githubusercontent.com/awslabs/ecs-logs-collector/master/ecs-logs-collector.sh
```

3. Exécutez le script pour collecter les journaux et créer l'archive.

> **Note**
>
> Pour activer le mode de débogage pour le démon Docker et l'agent de conteneur Amazon ECS, ajoutez l'`--mode=enable-debug`option à la commande suivante. Cela peut redémarrer le démon Docker, qui tue tous les conteneurs exécutés sur l'instance. Pensez à drainer l'instance de conteneur et à transférer les tâches importantes vers d'autres instances de conteneur avant d'activer le mode de débogage. Pour de plus amples informations, veuillez consulter Vidange des instances de conteneurs Amazon ECS.

```
[ec2-user ~]$ sudo bash ./ecs-logs-collector.sh
```

> **Important**
>
> Nous vous recommandons de modifier les journaux et de supprimer toutes les données sensibles des fichiers. Vous pouvez rechercher des données connues, ainsi que des variables d'environnement telles que `AWS_ACCESS_KEY_IDAWS_SECRET_ACCESS_KEY`, et `AWS_SESSION_TOKEN` dans le fichier.

Une fois que vous avez exécuté le script, vous pouvez examiner les journaux collectés dans le dossier `collect` créé par le script. Le `collect.tgz` fichier est une archive compressée de tous les journaux, que vous pouvez partager AWS Support pour obtenir de l'aide au diagnostic.

Pour télécharger et exécuter le collecteur de journaux Amazon ECS pour Windows

1. Connectez-vous à votre instance de conteneur. Pour plus d'informations, consultez Connect to your Windows instance using RDP dans le guide de l' EC2 utilisateur Amazon.
2. Téléchargez le script du collecteur de journaux Amazon ECS à l'aide de PowerShell.

```
Invoke-WebRequest -OutFile ecs-logs-collector.ps1 https://
raw.githubusercontent.com/awslabs/aws-ecs-logs-collector-for-windows/master/ecs-
logs-collector.ps1
```

3. Exécutez le script pour collecter les journaux et créer l'archive.

> **Note**
>
> Pour activer le mode de débogage pour le démon Docker et l'agent de conteneur Amazon ECS, ajoutez l'`-RunMode debug`option à la commande suivante. Cette action redémarre le démon Docker, ce qui supprime tous les conteneurs qui s'exécutent sur l'instance. Pensez à drainer l'instance de conteneur et à transférer les tâches importantes vers d'autres instances de conteneur avant d'activer le mode de débogage. Pour de plus amples informations, veuillez consulter Vidange des instances de conteneurs Amazon ECS.

```
.\ecs-logs-collector.ps1
```

> **Important**
>
> Nous vous recommandons de modifier les journaux et de supprimer toutes les données sensibles des fichiers. Vous pouvez rechercher des données connues, ainsi que des variables d'environnement telles que `AWS_ACCESS_KEY_IDAWS_SECRET_ACCESS_KEY`, et `AWS_SESSION_TOKEN` dans le fichier.

Une fois que vous avez exécuté le script, vous pouvez examiner les journaux collectés dans le dossier `collect` créé par le script. Le `collect.tgz` fichier est une archive compressée de tous les journaux, que vous pouvez partager avec le AWS Support pour obtenir de l'aide au diagnostic.

Récupérez les informations de diagnostic d'Amazon ECS grâce à l'introspection de l'agent

L'API d'introspection de l'agent Amazon ECS fournit des informations sur l'état général de l'agent Amazon ECS et des instances de conteneur.

Vous pouvez utiliser l'API d'introspection de l'agent pour obtenir l'ID Docker d'un conteneur dans votre tâche. Vous pouvez aussi utiliser l'API d'introspection d'agent en vous connectant à une instance de conteneur à l'aide de SSH.

> **Important**
>
> Votre instance de conteneur doit avoir un rôle IAM qui autorise l'accès à Amazon ECS afin d'atteindre l'API d'introspection. Pour de plus amples informations, veuillez consulter Rôle IAM d'instance de conteneur Amazon ECS.

L'exemple suivant montre deux tâches, l'une en cours d'exécution et l'autre arrêtée.

> **Note**
>
> La commande suivante est redirigée vers le python -mjson.tool pour une meilleure lisibilité.

```
curl http://localhost:51678/v1/tasks | python -mjson.tool
```

Sortie :

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1095  100  1095    0     0   117k      0 --:--:-- --:--:-- --:--:--  133k
{
    "Tasks": [
        {
            "Arn": "arn:aws:ecs:us-west-2:aws_account_id:task/090eff9b-1ce3-4db6-848a-a8d14064fd24",
            "Containers": [
                {
```

```
                    "DockerId":
 "189a8ff4b5f04affe40e5160a5ffadca395136eb5faf4950c57963c06f82c76d",
                    "DockerName": "ecs-console-sample-app-static-6-simple-
app-86caf9bcabe3e9c61600",
                    "Name": "simple-app"
                },
                {
                    "DockerId":
 "f7f1f8a7a245c5da83aa92729bd28c6bcb004d1f6a35409e4207e1d34030e966",
                    "DockerName": "ecs-console-sample-app-static-6-busybox-
ce83ce978a87a890ab01",
                    "Name": "busybox"
                }
            ],
            "Family": "console-sample-app-static",
            "KnownStatus": "STOPPED",
            "Version": "6"
        },
        {
            "Arn": "arn:aws:ecs:us-west-2:aws_account_id:task/1810e302-eaea-4da9-
a638-097bea534740",
            "Containers": [
                {
                    "DockerId":
 "dc7240fe892ab233dbbcee5044d95e1456c120dba9a6b56ec513da45c38e3aeb",
                    "DockerName": "ecs-console-sample-app-static-6-simple-app-
f0e5859699a7aecfb101",
                    "Name": "simple-app"
                },
                {
                    "DockerId":
 "096d685fb85a1ff3e021c8254672ab8497e3c13986b9cf005cbae9460b7b901e",
                    "DockerName": "ecs-console-sample-app-static-6-
busybox-92e4b8d0ecd0cce69a01",
                    "Name": "busybox"
                }
            ],
            "DesiredStatus": "RUNNING",
            "Family": "console-sample-app-static",
            "KnownStatus": "RUNNING",
            "Version": "6"
        }
    ]
```

```
}
```

Dans l'exemple précédent, la tâche arrêtée (*090eff9b-1ce3-4db6-848a-a8d14064fd24*) possède deux conteneurs. Vous pouvez utiliser docker inspect ***container-ID*** pour afficher des informations détaillées sur chaque conteneur. Pour de plus amples informations, veuillez consulter Introspection des conteneurs Amazon ECS.

Diagnostic Docker dans Amazon ECS

Docker fournit plusieurs outils de diagnostic qui peuvent vous aider à résoudre les problèmes que vous rencontrez avec vos conteneurs et vos tâches. Pour plus d'informations sur tous les utilitaires de ligne de commande Docker disponibles, consultez la référence de la CLI Docker dans la documentation Docker. Vous pouvez accéder aux utilitaires de ligne de commande Docker en vous connectant à une instance de conteneur à l'aide de SSH.

Les codes de sortie fournis par les conteneurs Docker peuvent également offrir des informations de diagnostic (par exemple, le code de sortie 137 signifie que le conteneur a reçu un signal `SIGKILL`). Pour plus d'informations, consultez la rubrique Exit Status de la documentation Docker.

Répertorier les conteneurs Docker dans Amazon ECS

Vous pouvez utiliser la commande docker ps de votre instance de conteneur pour dresser la liste des conteneurs en cours d'exécution. Dans l'exemple suivant, seul l'agent de conteneur Amazon ECS est en cours d'exécution. Pour plus d'informations, consultez la rubrique docker ps de la documentation Docker.

```
docker ps
```

Sortie :

```
CONTAINER ID        IMAGE                               COMMAND             CREATED
      STATUS              PORTS                         NAMES
cee0d6986de0        amazon/amazon-ecs-agent:latest   "/agent"            22 hours ago
      Up 22 hours         127.0.0.1:51678->51678/tcp   ecs-agent
```

Vous pouvez utiliser la commande docker ps -a pour afficher tous les conteneurs (même ceux qui ont été arrêtés ou supprimés). Cela peut s'avérer utile pour répertorier les conteneurs qui s'arrêtent

inopinément. Dans l'exemple suivant, le conteneur `f7f1f8a7a245` a disparu il y a 9 secondes. Il n'est donc pas visible dans une sortie docker ps sans l'indicateur `-a`.

```
docker ps -a
```

Sortie :

```
CONTAINER ID        IMAGE                                          COMMAND
 CREATED             STATUS                          PORTS                          NAMES
db4d48e411b1        amazon/ecs-emptyvolume-base:autogenerated   "not-applicable"
  19 seconds ago                                                                    ecs-
console-sample-app-static-6-internalecs-emptyvolume-source-c09288a6b0cba8a53700
f7f1f8a7a245        busybox:buildroot-2014.02                   "\"sh -c '/bin/sh -c
  22 hours ago        Exited (137) 9 seconds ago                                    ecs-
console-sample-app-static-6-busybox-ce83ce978a87a890ab01
189a8ff4b5f0        httpd:2                                     "httpd-foreground"
  22 hours ago        Exited (137) 40 seconds ago                                   ecs-
console-sample-app-static-6-simple-app-86caf9bcabe3e9c61600
0c7dca9321e3        amazon/ecs-emptyvolume-base:autogenerated   "not-applicable"
  22 hours ago                                                                      ecs-
console-sample-app-static-6-internalecs-emptyvolume-source-90fefaa68498a8a80700
cee0d6986de0        amazon/amazon-ecs-agent:latest              "/agent"
  22 hours ago        Up 22 hours                   127.0.0.1:51678->51678/tcp   ecs-
agent
```

Afficher les journaux Docker dans Amazon ECS

Vous pouvez consulter les flux `STDOUT` et `STDERR` d'un conteneur avec la commande docker logs. Dans cet exemple, les journaux sont affichés pour le *`dc7240fe892a`* conteneur et transmis via la head commande par souci de concision. Pour plus d'informations, consultez docker logs dans la documentation Docker.

> **Note**
>
> Les journaux Docker ne sont disponibles que sur l'instance de conteneur si vous utilisez le pilote de journal `json` par défaut. Si vous avez configuré vos tâches pour utiliser le pilote de `awslogs` journal, les journaux de vos conteneurs sont disponibles dans CloudWatch Logs. Pour de plus amples informations, veuillez consulter Envoyez les journaux Amazon ECS à CloudWatch .

```
docker logs dc7240fe892a | head
```

Sortie :

```
AH00558: httpd: Could not reliably determine the server's fully qualified domain name,
 using 172.17.0.11. Set the 'ServerName' directive globally to suppress this message
AH00558: httpd: Could not reliably determine the server's fully qualified domain name,
 using 172.17.0.11. Set the 'ServerName' directive globally to suppress this message
[Thu Apr 23 19:48:36.956682 2015] [mpm_event:notice] [pid 1:tid 140327115417472]
 AH00489: Apache/2.4.12 (Unix) configured -- resuming normal operations
[Thu Apr 23 19:48:36.956827 2015] [core:notice] [pid 1:tid 140327115417472] AH00094:
 Command line: 'httpd -D FOREGROUND'
10.0.1.86 - - [23/Apr/2015:19:48:59 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:48:59 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:49:28 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:49:29 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:49:50 +0000] "-" 408 -
10.0.0.154 - - [23/Apr/2015:19:49:50 +0000] "-" 408 -
10.0.1.86 - - [23/Apr/2015:19:49:58 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:49:59 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:50:28 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:50:29 +0000] "GET / HTTP/1.1" 200 348
time="2015-04-23T20:11:20Z" level="fatal" msg="write /dev/stdout: broken pipe"
```

Inspectez les conteneurs Docker dans Amazon ECS

Si vous avez l'ID Docker d'un conteneur, vous pouvez l'examiner avec la commande docker inspect. L'inspection des conteneurs offre la vue la plus détaillée de l'environnement dans lequel un conteneur a été lancé. Pour plus d'informations, consultez la rubrique docker inspect de la documentation Docker.

```
docker inspect dc7240fe892a
```

Sortie :

```
[{
    "AppArmorProfile": "",
    "Args": [],
    "Config": {
        "AttachStderr": false,
        "AttachStdin": false,
```

```
        "AttachStdout": false,
        "Cmd": [
            "httpd-foreground"
        ],
        "CpuShares": 10,
        "Cpuset": "",
        "Domainname": "",
        "Entrypoint": null,
        "Env": [
            "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/
local/apache2/bin",
            "HTTPD_PREFIX=/usr/local/apache2",
            "HTTPD_VERSION=2.4.12",
            "HTTPD_BZ2_URL=https://www.apache.org/dist/httpd/httpd-2.4.12.tar.bz2"
        ],
        "ExposedPorts": {
            "80/tcp": {}
        },
        "Hostname": "dc7240fe892a",
...
```

Configuration de la sortie détaillée du démon Docker dans Amazon ECS

Si vous rencontrez des problèmes avec les conteneurs ou les images Docker, vous pouvez activer le mode de débogage sur votre daemon Docker. L'utilisation du débogage fournit une sortie plus détaillée du démon. Vous pouvez l'utiliser pour récupérer les messages d'erreur envoyés depuis des registres de conteneurs, tels qu'Amazon ECR.

> **Important**
>
> Cette procédure a été écrite pour l'AMI Amazon Linux optimisée pour Amazon ECS. Pour les autres systèmes d'exploitation, consultez la section Activer le débogage, contrôler et configurer Docker avec systemd dans la documentation Docker.

Pour utiliser le mode de débogage du démon Docker sur l'AMI Amazon Linux optimisée pour Amazon ECS

1. Connectez-vous à votre instance de conteneur.

2. Ouvrez le fichier d'options Docker dans un éditeur de texte, tel que vi. Pour l'AMI Amazon Linux optimisée pour Amazon ECS, le fichier d'options Docker se trouve sous `/etc/sysconfig/docker`.
3. Recherchez l'instruction d'options Docker, puis ajoutez l'option `-D` à la chaîne, à l'intérieur des guillemets.

> **Note**
>
> Si l'instruction d'options Docker commence par le signe #, supprimez ce caractère pour supprimer la mise en commentaire de l'instruction et activer les options.

Pour l'AMI optimisée pour Amazon ECS, l'instruction d'options Docker s'appelle `OPTIONS`. Par exemple :

```
# Additional startup options for the Docker daemon, for example:
# OPTIONS="--ip-forward=true --iptables=true"
# By default we limit the number of open files per container
OPTIONS="-D --default-ulimit nofile=1024:4096"
```

4. Enregistrez le fichier et quittez votre éditeur de texte.
5. Redémarrez le démon Docker.

```
sudo service docker restart
```

La sortie est la suivante :

```
Stopping docker:                                           [  OK  ]
Starting docker: .                                   [  OK  ]
```

6. Redémarrez l'agent Amazon ECS.

```
sudo service ecs restart
```

Vos journaux Docker doivent désormais présenter des informations plus détaillées.

```
time="2015-12-30T21:48:21.907640838Z" level=debug msg="Unexpected response from
 server: \"{\\\"errors\\\":[{\\\"code\\\":\\\"DENIED\\\",\\\"message\\\":\\\"User:
```

```
 arn:aws:sts::1111:assumed-role/ecrReadOnly/i-abcdefg is not authorized to perform:
 ecr:InitiateLayerUpload on resource: arn:aws:ecr:us-east-1:1111:repository/nginx_test
\\\"}]}\\n\" http.Header{\"Connection\":[]string{\"keep-alive\"}, \"Content-Type\":
[]string{\"application/json; charset=utf-8\"}, \"Date\":[]string{\"Wed, 30 Dec 2015
 21:48:21 GMT\"}, \"Docker-Distribution-Api-Version\":[]string{\"registry/2.0\"},
 \"Content-Length\":[]string{\"235\"}}"
```

Résoudre les problèmes liés au Docker dans `API error (500): devmapper` Amazon ECS

L'erreur Docker suivante indique que le stockage par groupe mince sur votre instance de conteneur est saturé et que le démon Docker ne peut pas créer de nouveaux conteneurs :

```
CannotCreateContainerError: API error (500): devmapper: Thin Pool has 4350 free data
 blocks which is less than minimum required 4454 free data blocks. Create more free
 space in thin pool or use dm.min_free_space option to change behavior
```

Par défaut, Amazon Linux optimisé pour Amazon ECS, AMIs à partir des versions `2015.09.d` et versions ultérieures, est lancé avec un volume de 8 Go pour le système d'exploitation connecté `/dev/xvda` et monté en tant que racine du système de fichiers. Un volume supplémentaire de 22 Gio est attaché à `/dev/xvdcz`, qui est utilisé par Docker pour le stockage des métadonnées et des images. Si cet espace de stockage est saturé, le démon Docker ne peut pas créer de nouveaux conteneurs.

La manière la plus simple d'ajouter du stockage à vos instances de conteneur est de résilier les instances existantes et d'en lancer de nouvelles avec de plus grands volumes de stockage de données. Toutefois, si vous ne pouvez pas procéder de cette façon, vous pouvez ajouter du stockage au groupe de volumes utilisé par Docker et étendre son volume logique en suivant les procédures indiquées dans Linux optimisé pour Amazon ECS AMIs.

Si votre stockage d'instance de conteneur se remplit trop vite, vous pouvez limiter cela de la façon suivante :

- Pour afficher les informations par groupe mince, exécutez la commande suivante sur votre instance de conteneur :

  ```
  docker info
  ```

- (Amazon ECS container agent 1.8.0 et versions ultérieures) Vous pouvez réduire le temps pendant lequel les conteneurs arrêtés ou sortis restent sur vos instances de conteneur. La variable de configuration de l'agent `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` définit la durée d'attente entre l'arrêt d'une tâche et la suppression du conteneur Docker (par défaut, cette valeur est de 3 heures). Cette opération supprime les données du conteneur Docker. Si cette valeur est trop faible, il se peut que vous ne puissiez pas inspecter vos conteneurs arrêtés ou consulter les journaux avant leur suppression. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.
- Vous pouvez supprimer les conteneurs non actifs et les images inutilisées de vos instances de conteneur. Vous pouvez utiliser les exemples de commande suivants pour supprimer manuellement les conteneurs arrêtés et les images inutilisées. Les conteneurs supprimés ne peuvent pas être examinés par la suite et les images supprimées doivent être extraites à nouveau avant d'être utilisées pour lancer de nouveaux conteneurs.

  Pour supprimer les conteneurs qui ne sont pas en cours d'exécution, exécutez la commande suivante sur votre instance de conteneur :

  ```
  docker rm $(docker ps -aq)
  ```

  Pour supprimer les images non utilisées, exécutez la commande suivante sur votre instance de conteneur :

  ```
  docker rmi $(docker images -q)
  ```

- Vous pouvez supprimer les blocs de données inutilisés dans les conteneurs. Vous pouvez utiliser la commande suivante pour exécuter fstrim sur un conteneur en cours d'exécution et ignorer les blocs de données inutilisés par le système de fichiers du conteneur.

  ```
  sudo sh -c "docker ps -q | xargs docker inspect --format='{{ .State.Pid }}' | xargs -
  IZ fstrim /proc/Z/root/"
  ```

Résoudre les problèmes liés à Amazon ECS Exec

Voici des notes de résolution pour vous aider à diagnostiquer d'où viennent les erreurs lorsque vous utilisez ECS Exec.

Vérifiez à l'aide de l'Exec Checker

Le script ECS Exec Checker permet de vérifier et de valider que votre cluster et votre tâche Amazon ECS répondent aux conditions requises pour utiliser la fonctionnalité ECS Exec. Le script ECS Exec Checker vérifie à la fois votre AWS CLI environnement et votre cluster et vérifie que les tâches sont prêtes pour ECS Exec, en appelant divers APIs en votre nom. L'outil nécessite la dernière version du AWS CLI et doit être `jq` disponible. Pour plus d'informations, consultez ECS Exec Checker activé. GitHub

Erreur lors de l'appel à **`execute-command`**

Si une erreur `The execute command failed` se produit, les éléments suivants peuvent en être à l'origine.

- La tâche ne dispose pas des autorisations requises. Vérifiez que la définition de tâche utilisée pour lancer votre tâche a un rôle IAM de tâche défini et que le rôle dispose des autorisations requises. Pour de plus amples informations, veuillez consulter Autorisations ECS Exec.
- L'agent SSM n'est pas installé ou n'est pas en cours d'exécution.
- Il existe une interface Amazon VPC endpoint pour Amazon ECS, mais il n'y en a pas une pour Systems Manager Session Manager.

Résoudre les problèmes liés à Amazon ECS Anywhere

Amazon ECS Anywhere fournit une assistance pour l'enregistrement d'une instance externe telle qu'un serveur sur site ou une machine virtuelle (VM) sur votre cluster Amazon ECS. Voici les problèmes courants que vous pouvez rencontrer et les recommandations générales pour leur résolution.

Rubriques

- Problèmes d'enregistrement d'instance externe
- Problèmes de réseau d'instance externe
- Problèmes d'exécution de tâches sur votre instance externe

Problèmes d'enregistrement d'instance externe

Lorsque vous enregistrez une instance externe auprès de votre cluster Amazon ECS, les conditions suivantes doivent être remplies :

- Une AWS Systems Manager activation, qui consiste en un ID d'activation et un code d'activation, doit être récupérée. Vous l'utilisez pour enregistrer l'instance externe en tant qu'instance gérée par Systems Manager. Lorsqu'une activation de Systems Manager est demandée, spécifiez une limite d'enregistrement et une date d'expiration. La limite d'enregistrement spécifie le nombre maximal d'instances qui peuvent être enregistrées à l'aide de l'activation. La valeur par défaut pour la limite d'enregistrement est 1 instance. La date d'expiration spécifie la date à laquelle l'activation expire. La valeur par défaut est 24 heures. Si l'activation de Systems Manager que vous utilisez pour enregistrer votre instance externe n'est pas valide, demandez-en une nouvelle. Pour de plus amples informations, veuillez consulter Enregistrement d'une instance externe dans un cluster Amazon ECS.
- Une politique IAM est utilisée pour fournir à votre instance externe les autorisations dont elle a besoin pour communiquer avec les opérations AWS d'API. Si cette stratégie gérée n'est pas créée correctement et ne contient pas les autorisations requises, l'enregistrement d'instance externe échoue. Pour de plus amples informations, veuillez consulter Rôle IAM dans Amazon ECS Anywhere.
- Amazon ECS fournit un script d'installation qui installe Docker, l'agent de conteneur Amazon ECS et le Systems Manager Agent sur votre instance externe. Si le script d'installation échoue, il est probable qu'il ne puisse plus être exécuté sur la même instance sans qu'une erreur ne se produise. Dans ce cas, suivez le processus de nettoyage pour effacer les AWS ressources de l'instance afin de pouvoir réexécuter le script d'installation. Pour de plus amples informations, veuillez consulter Annulation de l'enregistrement d'une instance externe Amazon ECS.

> **Note**
>
> Sachez que si le script d'installation a demandé et utilisé avec succès l'activation de Systems Manager, toute autre exécution du script d'installation utilise à nouveau l'activation de Systems Manager. Cela peut à son tour vous amener à atteindre la limite d'enregistrement pour cette activation. Si cette limite est atteinte, vous devez recréer une activation.

- Lors de l'exécution du script d'installation sur une instance externe pour les charges de travail du GPU, si le pilote NVIDIA n'est pas détecté ou configuré correctement, une erreur se produit. Le script d'installation utilise la commande `nvidia-smi` pour confirmer l'existence du pilote NVIDIA.

Problèmes de réseau d'instance externe

Pour communiquer toute modification, votre instance externe nécessite une connexion réseau à AWS. Si votre instance externe perd sa connexion réseau AWS, les tâches exécutées sur vos instances continuent de s'exécuter de toute façon, sauf si elles sont arrêtées manuellement. Une fois la connexion rétablie, les AWS informations d'identification utilisées par l'agent de conteneur Amazon ECS et l'agent Systems Manager sur l'instance externe sont renouvelées automatiquement. AWS Pour plus d'informations sur les AWS domaines utilisés pour la communication entre votre instance externe et AWS, consultezRéseaux .

Problèmes d'exécution de tâches sur votre instance externe

Si vos tâches ou conteneurs ne parviennent pas à s'exécuter sur votre instance externe, cela est généralement dû au réseau ou aux autorisations. Si vos conteneurs extraient leurs images d'Amazon ECR ou sont configurés pour envoyer des journaux de conteneurs à Logs, votre définition de tâche doit spécifier un rôle IAM d'exécution de tâche valide. CloudWatch Sans un rôle IAM d'exécution de tâche valide, vos conteneurs ne démarrent pas. Pour en savoir plus sur les problèmes liés au réseau, consultez Problèmes de réseau d'instance externe.

> **Important**
>
> Amazon ECS fournit l'outil de collecte des journaux Amazon ECS. Vous pouvez l'utiliser pour collecter des journaux de vos instances externes à des fins de dépannage. Pour de plus amples informations, veuillez consulter Collecte des journaux de conteneurs avec le collecteur de journaux Amazon ECS .

AWS Fargate limitation des quotas

AWS Fargate limite les taux de lancement des tâches Amazon ECS et des pods Amazon EKS à des quotas (anciennement appelés limites) à l'aide d'un algorithme de bucket à jetons pour chaque AWS compte, région par région. Avec cet algorithme, votre compte dispose d'un compartiment contenant un nombre spécifique de jetons. Le nombre de jetons dans le compartiment représente votre quota

de taux à une seconde donnée. Chaque compte client dispose d'un compartiment de jetons de tâches et de pods qui s'épuise en fonction du nombre de tâches et de pods lancés par le compte client. Ce compartiment de jetons affiche un maximum de compartiment qui vous permet d'effectuer périodiquement un nombre plus élevé de demandes, et un taux de recharge qui vous permet de maintenir un taux de demande stable aussi longtemps que nécessaire.

Par exemple, la taille du compartiment de jetons de tâches et de pods pour un compte client Fargate est de 100 jetons et le taux de recharge est de 20 jetons par seconde. Par conséquent, vous pouvez lancer immédiatement jusqu'à 100 tâches Amazon ECS et pods Amazon EKS par compte client, avec un taux de lancement soutenu de 20 tâches Amazon ECS et pods Amazon EKS par seconde.

Actions	Capacité maximale du compartiment (ou taux en rafale)	Taux de recharge du compartiment (ou taux soutenu)
Quota de taux de ressources Fargate pour les tâches Amazon ECS à la demande et les pods Amazon EKS[1]	100	20
Quota de taux de ressources Fargate pour les tâches Spot Amazon ECS	100	20

[1]Les comptes qui lancent uniquement des pods Amazon EKS ont un taux en rafale de 20 avec un taux de lancement soutenu de 20 pods par seconde lors de l'utilisation des versions de plateforme décrites dans les Versions de la plateforme Amazon EKS.

Limiter l'`RunTask`API dans Fargate

En outre, Fargate limite le taux de requêtes lors du lancement de tâches à l'aide de l'API `RunTask` d'Amazon ECS à l'aide d'un quota distinct. Fargate limite les demandes d'API `RunTask` Amazon ECS AWS pour chaque compte par région. Chaque requête que vous effectuez supprime un jeton du compartiment. Le but est de favoriser la performance du service et d'assurer une utilisation équitable pour tous les clients Fargate. Les appels d'API sont soumis aux quotas de requêtes, qu'ils proviennent de la console Amazon Elastic Container Service, d'un outil de ligne de commande ou d'une application tierce. Le quota de taux pour les appels à l'API `RunTask` d'Amazon ECS est de 20 appels par seconde (rafale et soutenu). Chaque appel à cette API peut cependant lancer jusqu'à 10 tâches. Cela signifie que vous pouvez lancer 100 tâches en une seconde en effectuant 10 appels

à cette API, demandant que 10 tâches soient lancées dans chaque appel. De même, vous pouvez également effectuer 20 appels à cette API, demandant que 5 tâches soient lancées dans chaque appel. Pour plus d'informations sur la limitation des API pour l'`RunTask`API Amazon ECS, consultez la section Limitation des demandes d'API dans le manuel Amazon ECS API Reference.

Dans la pratique, les taux de lancement des tâches et des pods dépendent également d'autres considérations telles que les images de conteneur à télécharger et à décompresser, les surveillances de l'état et d'autres intégrations activées, telles que l'enregistrement des tâches ou des pods auprès d'un équilibreur de charge. Les clients constatent des variations dans les taux de lancement des tâches et des modules par rapport aux quotas présentés précédemment, en fonction des fonctionnalités activées par les clients.

Ajustement des quotas tarifaires dans Fargate

Vous pouvez demander une augmentation des quotas de limitation de taux pour votre compte AWS . Pour plus d'informations, consultez Demande d'augmentation de quotas dans le Guide de l'utilisateur Quotas de service .

Raisons de l'échec de l'API Amazon ECS

Lorsqu'une action d'API que vous avez déclenchée par le biais de l'API Amazon ECS, de la console ou d'Amazon AWS CLI ECS se termine par un message `failures` d'erreur, les informations suivantes peuvent vous aider à résoudre le problème. L'échec renvoie une raison et l'Amazon Resource Name (ARN) de la ressource associée à l'échec.

De nombreuses ressources sont spécifiques à la région. Lorsque vous utilisez la console, assurez-vous donc de définir la région correspondant à vos ressources. Lorsque vous utilisez le AWS CLI, assurez-vous que vos AWS CLI commandes sont envoyées à la bonne région avec le `--region` *`region`* paramètre.

Pour plus d'informations sur la structure du type de données `Failure`, consultez Failure (Échec) dans la Référence d'API Amazon Elastic Container Service.

Voici des exemples de messages d'échec que vous pouvez recevoir lors de l'exécution de commandes d'API.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
`DescribeClusters`	`MISSING`	Le cluster spécifié n'a pas été trouvé. Vérifiez l'orthographe du nom du cluster.
`DescribeInstances`	`MISSING`	L'instance de conteneur spécifiée n'a pas été trouvée. Vérifiez que vous avez spécifié le cluster dans lequel l'instance de conteneur est enregistrée et que l'ARN ou l'ID de l'instance de conteneur est correct.
`DescribeServices`	`MISSING`	Le service spécifié n'a pas été trouvé. Vérifiez que la région ou le cluster correct est spécifié et que l'ARN ou le nom du service est valide.
`DescribeTasks`	`MISSING`	La tâche spécifiée n'a pas été trouvée. Vérifiez que la région ou le cluster correct est spécifié et que l'ID ou l'ARN de la tâche est valide.
`DescribeTasks`	`TaskFailedToStart: RESOURCE:*`	En cas d'`RESOURCE: CPU` erreur, le nombre de CPUs requêtes demandées par la tâche n'est pas disponibl e sur vos instances de conteneur. Cela se produit généralement lorsque l'unité de processeur requise dans la définition de votre tâche est supérieure à la taille

Action d’API	Motif de l'échec ou de l'arrêt	Cause
		du processeur des EC2 instances Amazon définies dans le groupe Auto Scaling mappé au fournisseur de capacité. Vous devez vérifier la configuration de votre fournisseur de capacité. En cas d'erreur `RESOURCE:MEMORY` , la quantité de mémoire demandée par la tâche n'est pas disponible sur vos instances de conteneur . Cela se produit généralement lorsque la quantité de mémoire requise dans votre définition de tâche est supérieure à la mémoire prise en charge sur les EC2 instances Amazon définies dans le groupe Auto Scaling mappé au fournisseur de capacité. Vous devez vérifier la configuration de votre fournisseur de capacité.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
	`TaskFailedToStart: AGENT`	L'instance de conteneur avec laquelle vous avez essayé de lancer une tâche comporte un agent qui est actuellem ent déconnecté. Afin d'éviter de longs délais d'attente pour le placement de la tâche, la demande a été rejetée. Pour plus d'informations sur la résolution problèmes des agents déconnectés, consultez Comment résoudre les problèmes liés à un agent Amazon ECS déconnecté ?.
	`TaskFailedToStart: MemberOf placement constraint unsatisfi ed`	Aucune instance de conteneur ne répond aux contraintes de placement définies dans votre définition de tâche.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
	`TaskFailedToStart: ATTRIBUTE`	La définition de votre tâche contient un paramètre nécessitant un attribut d'instance de conteneur spécifique qui n'est pas disponible sur vos instances de conteneur. Cela se produit, par exemple, si votre tâche utilise le mode réseau `awsvpc`, mais qu'aucune instance dans les sous-réseaux spécifiés ne comporte l'attribut `ecs.capability.task-eni` . Pour plus d'informations sur les attributs requis pour des paramètres de définition de tâche et des variables de configuration d'agent spécifiques, consultez Paramètres de définition des tâches Amazon ECS pour le type de lancement Fargate et Configuration de l'agent de conteneur Amazon ECS.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
	`TaskFailedToStart: NO ACTIVE INSTANCES`	Il n'y a aucune instance active dans votre fournisseur de capacité. Pour plus d'informations sur la gestion de vos groupes Auto Scaling, consultez la section Groupes Auto Scaling dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling.
	`TaskFailedToStart: EMPTY CAPACITY PROVIDER`	Il n'existe pas d'instances dans votre cluster. Cela est probablement dû à un fournisseur de capacité vide ou au fait que les instances du fournisseur de capacité ne sont pas enregistrées auprès du cluster. Pour plus d'informations sur la gestion de vos groupes Auto Scaling, consultez la section Groupes Auto Scaling dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling.
`GetTaskProtection`	`MISSING`	La tâche spécifiée n'a pas été trouvée. Vérifiez que l'ARN ou le nom du cluster et l'ARN ou l'ID de tâche sont valides.

Action d’API	Motif de l'échec ou de l'arrêt	Cause
	`TASK_NOT_VALID`	La tâche spécifiée ne fait pas partie d'un service Amazon ECS. Seules les tâches gérées par les services Amazon ECS peuvent être protégées. Vérifiez l'ARN ou l'ID de la tâche et réessayez.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
`RunTask` ou `StartTask`	`RESOURCE:*`	La ou les ressources demandées par la tâche ne sont pas disponibles sur les instances de conteneur du cluster. Si la ressource demandée concerne le CPU, la mémoire, les ports ou les interfaces réseau Elastic, vous devrez peut-être ajouter d'autres instances de conteneur à votre cluster. Les erreurs `RESOURCE:ENI` indiquent que votre cluster ne dispose pas de points d'attache d'interface réseau Elastic, qui sont requis pour les tâches utilisant le mode réseau `awsvpc`. Le nombre d'interfaces réseau pouvant être associées aux EC2 instances Amazon est limité, et l'interface réseau principale compte pour une seule. Pour plus d'informations sur le nombre d'interfaces réseau prises en charge pour chaque type d'instance, consultez la section Adresses IP par interface réseau et par type d'instance dans le guide de EC2 l'utilisateur Amazon. En cas d'`RESOURCE:GPU` erreur, le nombre de

Action d'API	Motif de l'échec ou de l'arrêt	Cause
		GPUs requêtes demandées par la tâche n'est pas disponible et vous devrez peut-être ajouter des instances de conteneur compatibles GPU à votre cluster. Pour de plus amples informations, veuillez consulter Définitions de tâches Amazon ECS pour les charges de travail du GPU.
	`AGENT`	L'instance de conteneur avec laquelle vous avez essayé de lancer une tâche comporte un agent qui est actuellement déconnecté. Afin d'éviter de longs délais d'attente pour le placement de la tâche, la demande a été rejetée. Pour plus d'informations sur la résolution problèmes des agents déconnectés, consultez Comment résoudre les problèmes liés à un agent Amazon ECS déconnecté ?.
	`LOCATION`	L'instance de conteneur sur laquelle vous avez tenté de lancer une tâche se trouve dans une zone de disponibilité différente de celle du ou des sous-réseaux que vous avez spécifiés dans votre `awsVpcConfiguration` .

Action d'API	Motif de l'échec ou de l'arrêt	Cause
	`ATTRIBUTE`	La définition de votre tâche contient un paramètre nécessitant un attribut d'instance de conteneur spécifique qui n'est pas disponible sur vos instances de conteneur. Cela se produit, par exemple, si votre tâche utilise le mode réseau `awsvpc`, mais qu'aucune instance dans les sous-réseaux spécifiés ne comporte l'attribut `ecs.capability.task-eni` . Pour plus d'informations sur les attributs requis pour des paramètres de définition de tâche et des variables de configuration d'agent spécifiques, consultez Paramètres de définition des tâches Amazon ECS pour le type de lancement Fargate et Configuration de l'agent de conteneur Amazon ECS.
`StartTask`	`MISSING`	L'instance de conteneur sur laquelle vous avez tenté de lancer la tâche est introuvable. Vérifiez si le cluster ou la région spécifiés est incorrect ou si l'ARN ou l'ID de l'instance de conteneur est mal orthographié.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
	`INACTIVE`	L'enregistrement de l'instance de conteneur sur laquelle vous avez essayé de lancer une tâche a été précédemment annulé avec Amazon ECS et cette instance ne peut pas être utilisée.
`StopServiceDeployment`	`ECS deployment failed`	Un compte frauduleux exécutait l'`StopServiceDeployment` API.
`TagResource`	`InvalidParameterException`	L'ARN du service que vous balisez est au format court. Vous devez migrer vers le format long. Pour plus d'informations sur la façon de migrer l'ARN, consultez Migrer un ARN de service court Amazon ECS vers un ARN long.
`UpdateTaskProtection`	`DEPLOYMENT_BLOCKED`	Impossible de définir la protection des tâches car une ou plusieurs tâches protégées empêchent le déploiement du service d'atteindre un état stable. Désactivez la protection des tâches sur les tâches existantes ou attendez que la protection des tâches expire.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
	`MISSING`	La tâche spécifiée n'a pas été trouvée. Vérifiez que l'ARN ou le nom du cluster et l'ARN ou l'ID de tâche sont valides.
	`TASK_NOT_VALID`	La tâche spécifiée ne fait pas partie d'un service Amazon ECS. Seules les tâches gérées par les services Amazon ECS peuvent être protégées. Vérifiez l'ARN ou l'ID de la tâche et réessayez.

> **Note**
>
> Outre les scénarios d'échec décrits ici, les opérations d'API peuvent également échouer en raison d'exceptions, entraînant des réponses d'erreur. Pour obtenir la liste de ces exceptions, consultez la section Erreurs courantes.

Sécurité dans Amazon Elastic Container Service

La sécurité du cloud AWS est la priorité absolue. En tant que AWS client, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des entreprises les plus sensibles en matière de sécurité.

La sécurité est une responsabilité partagée entre vous AWS et vous. Le modèle de responsabilité partagée décrit cette notion par les termes sécurité du cloud et sécurité dans le cloud :

- Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute les AWS services dans le AWS cloud. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Des auditeurs tiers testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre des programmes de conformitéAWS. Pour en savoir plus sur les programmes de conformité qui s'appliquent à Amazon Elastic Container Service, veuillez consulter Services AWS concernés par le programme de conformité.
- Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris de la sensibilité de vos données, des exigences de votre entreprise, ainsi que de la législation et de la réglementation applicables.

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation d'Amazon ECS. Les rubriques suivantes vous montrent comment configurer Amazon ECS pour répondre à vos objectifs de sécurité et de conformité. Vous apprendrez également à utiliser d'autres AWS services qui vous aident à surveiller et à sécuriser vos ressources Amazon ECS.

Rubriques

- Gestion des identités et des accès pour Amazon Elastic Container Service
- Journalisation et surveillance dans Amazon Elastic Container Service
- Validation de la conformité pour Amazon Elastic Container Service
- AWS Fargate Norme fédérale de traitement de l'information (FIPS-140)
- Sécurité de l'infrastructure dans Amazon Elastic Container Service
- AWS modèle de responsabilité partagée pour Amazon ECS
- Bonnes pratiques en matière de sécurité réseau pour Amazon ECS
- Bonnes pratiques en matière de sécurité des tâches et des conteneurs Amazon ECS

Gestion des identités et des accès pour Amazon Elastic Container Service

AWS Identity and Access Management (IAM) est un outil Service AWS qui permet à un administrateur de contrôler en toute sécurité l'accès aux AWS ressources. Des administrateurs IAM contrôlent les personnes qui peuvent être authentifiées (connectées) et autorisées (disposant d'autorisations) à utiliser des ressources Amazon ECS. IAM est un Service AWS outil que vous pouvez utiliser sans frais supplémentaires.

Rubriques

- Public ciblé
- Authentification par des identités
- Gestion des accès à l'aide de politiques
- Fonctionnement d'Amazon Elastic Container Service avec IAM
- Exemples de politiques basées sur l'identité pour Amazon Elastic Container Service
- AWS politiques gérées pour Amazon Elastic Container Service
- Utilisation des rôles liés à un service pour Amazon ECS
- Rôles IAM pour Amazon ECS
- Autorisations requises pour la console Amazon ECS
- Autorisations requises pour les fonctions Lambda dans les déploiements Amazon ECS blue/green
- Autorisations IAM requises pour le dimensionnement automatique du service Amazon ECS
- Octroi de l'autorisation de baliser les ressources lors de la création
- Dépannage de l'identité et de l'accès Amazon Elastic Container Service
- Bonnes pratiques en matière d'IAM pour Amazon ECS

Public ciblé

La façon dont vous utilisez AWS Identity and Access Management (IAM) varie en fonction du travail que vous effectuez dans Amazon ECS.

Utilisateur du service : si vous utilisez le service Amazon ECS service pour accomplir votre tâche, votre administrateur vous fournira les informations d'identification et les autorisations

nécessaires. Vous pourrez avoir besoin d'autorisations supplémentaires si vous utilisez davantage de fonctions Amazon ECS. En comprenant bien la gestion des accès, vous saurez demander les autorisations appropriées à votre administrateur. Si vous ne pouvez pas accéder à une fonction dans Amazon ECS, veuillez consulter Dépannage de l'identité et de l'accès Amazon Elastic Container Service.

Administrateur du service : si vous êtes le responsable des ressources Amazon ECS de votre entreprise, vous bénéficiez probablement d'un accès total à ce service. C'est à vous de déterminer les fonctions et les ressources Amazon ECS auxquelles vos utilisateurs des services pourront accéder. Vous devez ensuite soumettre les demandes à votre administrateur IAM pour modifier les autorisations des utilisateurs de votre service. Consultez les informations sur cette page pour comprendre les concepts de base d'IAM. Pour découvrir la façon dont votre entreprise peut utiliser IAM avec Amazon ECS, veuillez consulter Fonctionnement d'Amazon Elastic Container Service avec IAM.

Administrateur IAM : si vous êtes un administrateur IAM, vous souhaiterez peut-être obtenir des informations sur la façon dont vous pouvez écrire des stratégies pour gérer l'accès à Amazon ECS. Pour afficher des exemples de stratégies basées sur l'identité Amazon ECS que vous pouvez utiliser dans IAM, veuillez consulter Exemples de politiques basées sur l'identité pour Amazon Elastic Container Service.

Authentification par des identités

L'authentification est la façon dont vous vous connectez à AWS l'aide de vos informations d'identification. Vous devez être authentifié (connecté à AWS) en tant qu'utilisateur IAM ou en assumant un rôle IAM. Utilisateur racine d'un compte AWS

Vous pouvez vous connecter en AWS tant qu'identité fédérée en utilisant les informations d'identification fournies par le biais d'une source d'identité. AWS IAM Identity Center Les utilisateurs (IAM Identity Center), l'authentification unique de votre entreprise et vos informations d'identification Google ou Facebook sont des exemples d'identités fédérées. Lorsque vous vous connectez avec une identité fédérée, votre administrateur aura précédemment configuré une fédération d'identités avec des rôles IAM. Lorsque vous accédez à AWS l'aide de la fédération, vous assumez indirectement un rôle.

Selon le type d'utilisateur que vous êtes, vous pouvez vous connecter au portail AWS Management Console ou au portail AWS d'accès. Pour plus d'informations sur la connexion à AWS, consultez la section Comment vous connecter à votre compte Compte AWS dans le guide de Connexion à AWS l'utilisateur.

Si vous y accédez AWS par programmation, AWS fournit un kit de développement logiciel (SDK) et une interface de ligne de commande (CLI) pour signer cryptographiquement vos demandes à l'aide de vos informations d'identification. Si vous n'utilisez pas d' AWS outils, vous devez signer vous-même les demandes. Pour plus d'informations sur l'utilisation de la méthode recommandée pour signer des demandes vous-même, consultez AWS Signature Version 4 pour les demandes d'API dans le Guide de l'utilisateur IAM.

Quelle que soit la méthode d'authentification que vous utilisez, vous devrez peut-être fournir des informations de sécurité supplémentaires. Par exemple, il vous AWS recommande d'utiliser l'authentification multifactorielle (MFA) pour renforcer la sécurité de votre compte. Pour plus d'informations, consultez Authentification multifactorielle dans le Guide de l'utilisateur AWS IAM Identity Center et Authentification multifactorielle AWS dans IAM dans le Guide de l'utilisateur IAM.

Compte AWS utilisateur root

Lorsque vous créez un Compte AWS, vous commencez par une identité de connexion unique qui donne un accès complet à toutes Services AWS les ressources du compte. Cette identité est appelée utilisateur Compte AWS root et est accessible en vous connectant avec l'adresse e-mail et le mot de passe que vous avez utilisés pour créer le compte. Il est vivement recommandé de ne pas utiliser l'utilisateur racine pour vos tâches quotidiennes. Protégez vos informations d'identification d'utilisateur racine et utilisez-les pour effectuer les tâches que seul l'utilisateur racine peut effectuer. Pour obtenir la liste complète des tâches qui vous imposent de vous connecter en tant qu'utilisateur racine, consultez Tâches nécessitant des informations d'identification d'utilisateur racine dans le Guide de l'utilisateur IAM.

Identité fédérée

La meilleure pratique consiste à obliger les utilisateurs humains, y compris ceux qui ont besoin d'un accès administrateur, à utiliser la fédération avec un fournisseur d'identité pour accéder à l'aide Services AWS d'informations d'identification temporaires.

Une identité fédérée est un utilisateur de l'annuaire des utilisateurs de votre entreprise, d'un fournisseur d'identité Web AWS Directory Service, du répertoire Identity Center ou de tout utilisateur qui y accède à l'aide des informations d'identification fournies Services AWS par le biais d'une source d'identité. Lorsque des identités fédérées y accèdent Comptes AWS, elles assument des rôles, qui fournissent des informations d'identification temporaires.

Pour une gestion des accès centralisée, nous vous recommandons d'utiliser AWS IAM Identity Center. Vous pouvez créer des utilisateurs et des groupes dans IAM Identity Center, ou vous pouvez

vous connecter et synchroniser avec un ensemble d'utilisateurs et de groupes dans votre propre source d'identité afin de les utiliser dans toutes vos applications Comptes AWS et applications. Pour obtenir des informations sur IAM Identity Center, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Utilisateurs et groupes IAM

Un utilisateur IAM est une identité au sein de vous Compte AWS qui possède des autorisations spécifiques pour une seule personne ou application. Dans la mesure du possible, nous vous recommandons de vous appuyer sur des informations d'identification temporaires plutôt que de créer des utilisateurs IAM ayant des informations d'identification à long terme telles que des mots de passe et des clés d'accès. Toutefois, si certains cas d'utilisation spécifiques nécessitent des informations d'identification à long terme avec les utilisateurs IAM, nous vous recommandons d'effectuer une rotation des clés d'accès. Pour plus d'informations, consultez Rotation régulière des clés d'accès pour les cas d'utilisation nécessitant des informations d'identification dans le Guide de l'utilisateur IAM.

Un groupe IAM est une identité qui concerne un ensemble d'utilisateurs IAM. Vous ne pouvez pas vous connecter en tant que groupe. Vous pouvez utiliser les groupes pour spécifier des autorisations pour plusieurs utilisateurs à la fois. Les groupes permettent de gérer plus facilement les autorisations pour de grands ensembles d'utilisateurs. Par exemple, vous pouvez nommer un groupe IAMAdminset lui donner les autorisations nécessaires pour administrer les ressources IAM.

Les utilisateurs sont différents des rôles. Un utilisateur est associé de manière unique à une personne ou une application, alors qu'un rôle est conçu pour être endossé par tout utilisateur qui en a besoin. Les utilisateurs disposent d'informations d'identification permanentes, mais les rôles fournissent des informations d'identification temporaires. Pour plus d'informations, consultez Cas d'utilisation pour les utilisateurs IAM dans le Guide de l'utilisateur IAM.

Rôles IAM

Un rôle IAM est une identité au sein de vous Compte AWS dotée d'autorisations spécifiques. Le concept ressemble à celui d'utilisateur IAM, mais le rôle IAM n'est pas associé à une personne en particulier. Pour assumer temporairement un rôle IAM dans le AWS Management Console, vous pouvez passer d'un rôle d'utilisateur à un rôle IAM (console). Vous pouvez assumer un rôle en appelant une opération d' AWS API AWS CLI ou en utilisant une URL personnalisée. Pour plus d'informations sur les méthodes d'utilisation des rôles, consultez Méthodes pour endosser un rôle dans le Guide de l'utilisateur IAM.

Les rôles IAM avec des informations d'identification temporaires sont utiles dans les cas suivants :

- Accès utilisateur fédéré : pour attribuer des autorisations à une identité fédérée, vous créez un rôle et définissez des autorisations pour le rôle. Quand une identité externe s'authentifie, l'identité est associée au rôle et reçoit les autorisations qui sont définies par celui-ci. Pour obtenir des informations sur les rôles pour la fédération, consultez Création d'un rôle pour un fournisseur d'identité tiers (fédération) dans le Guide de l'utilisateur IAM. Si vous utilisez IAM Identity Center, vous configurez un jeu d'autorisations. IAM Identity Center met en corrélation le jeu d'autorisations avec un rôle dans IAM afin de contrôler à quoi vos identités peuvent accéder après leur authentification. Pour plus d'informations sur les jeux d'autorisations, consultez Jeux d'autorisations dans le Guide de l'utilisateur AWS IAM Identity Center .
- Autorisations d'utilisateur IAM temporaires : un rôle ou un utilisateur IAM peut endosser un rôle IAM pour profiter temporairement d'autorisations différentes pour une tâche spécifique.
- Accès intercompte : vous pouvez utiliser un rôle IAM pour permettre à un utilisateur (principal de confiance) d'un compte différent d'accéder aux ressources de votre compte. Les rôles constituent le principal moyen d'accorder l'accès intercompte. Toutefois, dans certains Services AWS cas, vous pouvez associer une politique directement à une ressource (au lieu d'utiliser un rôle comme proxy). Pour en savoir plus sur la différence entre les rôles et les politiques basées sur les ressources pour l'accès intercompte, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.
- Accès multiservices — Certains Services AWS utilisent des fonctionnalités dans d'autres Services AWS. Par exemple, lorsque vous effectuez un appel dans un service, il est courant que ce service exécute des applications dans Amazon EC2 ou stocke des objets dans Amazon S3. Un service peut le faire en utilisant les autorisations d'appel du principal, un rôle de service ou un rôle lié au service.
  - Sessions d'accès direct (FAS) : lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, vous pouvez effectuer une action qui initie une autre action dans un autre service. FAS utilise les autorisations du principal appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur une politique lors de la formulation de demandes FAS, consultez Transmission des sessions d'accès.
  - Rôle de service : il s'agit d'un rôle IAM attribué à un service afin de réaliser des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer un rôle de service à partir d'IAM.

Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

- Rôle lié à un service — Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés à un service apparaissent dans votre Compte AWS répertoire et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

- Applications exécutées sur Amazon EC2 : vous pouvez utiliser un rôle IAM pour gérer les informations d'identification temporaires pour les applications qui s'exécutent sur une EC2 instance et qui envoient des demandes AWS CLI d' AWS API. Cela est préférable au stockage des clés d'accès dans l' EC2 instance. Pour attribuer un AWS rôle à une EC2 instance et le rendre disponible pour toutes ses applications, vous devez créer un profil d'instance attaché à l'instance. Un profil d'instance contient le rôle et permet aux programmes exécutés sur l' EC2 instance d'obtenir des informations d'identification temporaires. Pour plus d'informations, consultez Utiliser un rôle IAM pour accorder des autorisations aux applications exécutées sur des EC2 instances Amazon dans le guide de l'utilisateur IAM.

Gestion des accès à l'aide de politiques

Vous contrôlez l'accès en AWS créant des politiques et en les associant à AWS des identités ou à des ressources. Une politique est un objet AWS qui, lorsqu'il est associé à une identité ou à une ressource, définit leurs autorisations. AWS évalue ces politiques lorsqu'un principal (utilisateur, utilisateur root ou session de rôle) fait une demande. Les autorisations dans les politiques déterminent si la demande est autorisée ou refusée. La plupart des politiques sont stockées AWS sous forme de documents JSON. Pour plus d'informations sur la structure et le contenu des documents de politique JSON, consultez Vue d'ensemble des politiques JSON dans le Guide de l'utilisateur IAM.

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

Par défaut, les utilisateurs et les rôles ne disposent d'aucune autorisation. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent assumer les rôles.

Les politiques IAM définissent les autorisations d'une action, quelle que soit la méthode que vous utilisez pour exécuter l'opération. Par exemple, supposons que vous disposiez d'une politique qui autorise l'action `iam:GetRole`. Un utilisateur appliquant cette politique peut obtenir des informations sur le rôle à partir de AWS Management Console AWS CLI, de ou de l' AWS API.

Politiques basées sur l'identité

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l'utilisateur IAM.

Les politiques basées sur l'identité peuvent être classées comme des politiques en ligne ou des politiques gérées. Les politiques en ligne sont intégrées directement à un utilisateur, groupe ou rôle. Les politiques gérées sont des politiques autonomes que vous pouvez associer à plusieurs utilisateurs, groupes et rôles au sein de votre Compte AWS. Les politiques gérées incluent les politiques AWS gérées et les politiques gérées par le client. Pour découvrir comment choisir entre une politique gérée et une politique en ligne, consultez Choix entre les politiques gérées et les politiques en ligne dans le Guide de l'utilisateur IAM.

Politiques basées sur les ressources

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Par exemple, les politiques de confiance de rôle IAM et les politiques de compartiment Amazon S3 sont des politiques basées sur les ressources. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Les politiques basées sur les ressources sont des politiques en ligne situées dans ce service. Vous ne pouvez pas utiliser les politiques AWS gérées par IAM dans une stratégie basée sur les ressources.

Listes de contrôle d'accès (ACLs)

Les listes de contrôle d'accès (ACLs) contrôlent les principaux (membres du compte, utilisateurs ou rôles) autorisés à accéder à une ressource. ACLs sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Amazon S3 et AWS WAF Amazon VPC sont des exemples de services compatibles. ACLs Pour en savoir plus ACLs, consultez la présentation de la liste de contrôle d'accès (ACL) dans le guide du développeur Amazon Simple Storage Service.

Autres types de politique

AWS prend en charge d'autres types de politiques moins courants. Ces types de politiques peuvent définir le nombre maximum d'autorisations qui vous sont accordées par des types de politiques plus courants.

- Limite d'autorisations : une limite d'autorisations est une fonctionnalité avancée dans laquelle vous définissez le nombre maximal d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM (utilisateur ou rôle IAM). Vous pouvez définir une limite d'autorisations pour une entité. Les autorisations en résultant représentent la combinaison des politiques basées sur l'identité d'une entité et de ses limites d'autorisation. Les politiques basées sur les ressources qui spécifient l'utilisateur ou le rôle dans le champ `Principal` ne sont pas limitées par les limites d'autorisations. Un refus explicite dans l'une de ces politiques annule l'autorisation. Pour plus d'informations sur les limites d'autorisations, consultez Limites d'autorisations pour des entités IAM dans le Guide de l'utilisateur IAM.
- Politiques de contrôle des services (SCPs) : SCPs politiques JSON qui spécifient les autorisations maximales pour une organisation ou une unité organisationnelle (UO) dans AWS Organizations. AWS Organizations est un service permettant de regrouper et de gérer de manière centralisée Comptes AWS les multiples propriétés de votre entreprise. Si vous activez toutes les fonctionnalités d'une organisation, vous pouvez appliquer des politiques de contrôle des services (SCPs) à l'un ou à l'ensemble de vos comptes. Le SCP limite les autorisations pour les entités figurant dans les comptes des membres, y compris chacune Utilisateur racine d'un compte AWS d'entre elles. Pour plus d'informations sur les Organizations et consultez SCPs les politiques de contrôle des services dans le Guide de AWS Organizations l'utilisateur.
- Politiques de contrôle des ressources (RCPs) : RCPs politiques JSON que vous pouvez utiliser pour définir le maximum d'autorisations disponibles pour les ressources de vos comptes sans mettre à jour les politiques IAM associées à chaque ressource que vous possédez. Le RCP limite les autorisations pour les ressources des comptes membres et peut avoir un impact sur les

autorisations effectives pour les identités, y compris Utilisateur racine d'un compte AWS, qu'elles appartiennent ou non à votre organisation. Pour plus d'informations sur les Organizations RCPs, y compris une liste de ces Services AWS supports RCPs, consultez la section Resource control policies (RCPs) dans le guide de AWS Organizations l'utilisateur.

- Politiques de séance : les politiques de séance sont des politiques avancées que vous utilisez en tant que paramètre lorsque vous créez par programmation une séance temporaire pour un rôle ou un utilisateur fédéré. Les autorisations de séance en résultant sont une combinaison des politiques basées sur l'identité de l'utilisateur ou du rôle et des politiques de séance. Les autorisations peuvent également provenir d'une politique basée sur les ressources. Un refus explicite dans l'une de ces politiques annule l'autorisation. Pour plus d'informations, consultez Politiques de session dans le Guide de l'utilisateur IAM.

Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations en résultant sont plus compliquées à comprendre. Pour savoir comment AWS déterminer s'il faut autoriser une demande lorsque plusieurs types de politiques sont impliqués, consultez la section Logique d'évaluation des politiques dans le guide de l'utilisateur IAM.

Fonctionnement d'Amazon Elastic Container Service avec IAM

Avant d'utiliser IAM pour gérer l'accès à Amazon ECS, apprenez les fonctions IAM qui peuvent être utilisées dans cette situation.

Fonctionnalité IAM	Prise en charge d'Amazon ECS
Politiques basées sur l'identité	Oui
Politiques basées sur les ressources	Non
Actions de politique	Oui
Ressources de politique	Partielle
Clés de condition de politique	Oui
ACLs	Non

Fonctionnalité IAM	Prise en charge d'Amazon ECS
ABAC (étiquettes dans les politiques)	Oui
Informations d'identification temporaires	Oui
Transmission des sessions d'accès (FAS)	Oui
Rôles de service	Oui
Rôles liés à un service	Oui

Pour obtenir une vue d'ensemble de la façon dont Amazon ECS et les autres AWS services fonctionnent avec la plupart des fonctionnalités IAM, consultez les AWS services compatibles avec IAM dans le guide de l'utilisateur IAM.

Politiques basées sur l'identité pour Amazon ECS

Prend en charge les politiques basées sur l'identité : oui

Les politiques basées sur l'identité sont des documents de politique d'autorisations JSON que vous pouvez attacher à une identité telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d'actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l'utilisateur IAM.

Avec les politiques IAM basées sur l'identité, vous pouvez spécifier des actions et ressources autorisées ou refusées, ainsi que les conditions dans lesquelles les actions sont autorisées ou refusées. Vous ne pouvez pas spécifier le principal dans une politique basée sur une identité, car celle-ci s'applique à l'utilisateur ou au rôle auquel elle est attachée. Pour découvrir tous les éléments que vous utilisez dans une politique JSON, consultez Références des éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.

Exemples de politiques basées sur une identité pour Amazon ECS

Pour voir des exemples de politiques Amazon ECS basées sur l'identité, consultez Exemples de politiques basées sur l'identité pour Amazon Elastic Container Service.

Politiques basées sur les ressources au sein d'Amazon ECS

Prend en charge les politiques basées sur les ressources : non

Les politiques basées sur les ressources sont des documents de politique JSON que vous attachez à une ressource. Par exemple, les politiques de confiance de rôle IAM et les politiques de compartiment Amazon S3 sont des politiques basées sur les ressources. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou. Services AWS

Pour permettre un accès intercompte, vous pouvez spécifier un compte entier ou des entités IAM dans un autre compte en tant que principal dans une politique basée sur les ressources. L'ajout d'un principal intercompte à une politique basée sur les ressources ne représente qu'une partie de l'instauration de la relation d'approbation. Lorsque le principal et la ressource sont différents Comptes AWS, un administrateur IAM du compte sécurisé doit également accorder à l'entité principale (utilisateur ou rôle) l'autorisation d'accéder à la ressource. Pour ce faire, il attache une politique basée sur une identité à l'entité. Toutefois, si une politique basée sur des ressources accorde l'accès à un principal dans le même compte, aucune autre politique basée sur l'identité n'est requise. Pour plus d'informations, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Actions de politique pour Amazon ECS

Prend en charge les actions de politique : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Action` d'une politique JSON décrit les actions que vous pouvez utiliser pour autoriser ou refuser l'accès à une politique. Les actions de stratégie portent généralement le même nom que l'opération AWS d'API associée. Il existe quelques exceptions, telles que les actions avec autorisations uniquement qui n'ont pas d'opération API correspondante. Certaines opérations nécessitent également plusieurs actions dans une politique. Ces actions supplémentaires sont nommées actions dépendantes.

Intégration d'actions dans une politique afin d'accorder l'autorisation d'exécuter les opérations associées.

Pour afficher la liste des actions Amazon ECS, consultez Actions définies par Amazon Elastic Container Service dans Référence de l'autorisation de service.

Les actions de politique dans Amazon ECS utilisent le préfixe suivant avant l'action :

```
ecs
```

Pour indiquer plusieurs actions dans une seule déclaration, séparez-les par des virgules.

```
"Action": [
      "ecs:action1",
      "ecs:action2"
         ]
```

Vous pouvez aussi spécifier plusieurs actions à l'aide de caractères génériques (*). Par exemple, pour spécifier toutes les actions qui commencent par le mot `Describe`, incluez l'action suivante :

```
"Action": "ecs:Describe*"
```

Pour voir des exemples de politiques Amazon ECS basées sur l'identité, consultez Exemples de politiques basées sur l'identité pour Amazon Elastic Container Service.

Ressources de politique pour Amazon ECS

Supporte les ressources politiques : partielles

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément de politique JSON `Resource` indique le ou les objets auxquels l'action s'applique. Les instructions doivent inclure un élément `Resource` ou `NotResource`. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Vous pouvez le faire pour des actions qui prennent en charge un type de ressource spécifique, connu sous la dénomination autorisations de niveau ressource.

Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, telles que les opérations de liste, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Pour consulter la liste des types de ressources Amazon ECS et de leurs caractéristiques ARNs, consultez la section Ressources définies par Amazon Elastic Container Service dans le Service Authorization Reference. Pour connaître les actions avec lesquelles vous pouvez spécifier l'ARN de chaque ressource, consultez Actions définies par Amazon Elastic Container Service.

Certaines actions d'API Amazon ECS prennent en charge plusieurs ressources. Par exemple, plusieurs clusters peuvent être référencés lors de l'appel de l'action d'API `DescribeClusters`. Pour spécifier plusieurs ressources dans une seule instruction, séparez-les ARNs par des virgules.

```
"Resource": [
      "EXAMPLE-RESOURCE-1",
      "EXAMPLE-RESOURCE-2"
```

Par exemple, la ressource de cluster Amazon ECS a l'ARN suivant :

```
arn:${Partition}:ecs:${Region}:${Account}:cluster/${clusterName}
```

Pour spécifier `my-cluster-1` et `my-cluster-2` regrouper dans votre relevé, utilisez ce qui suit ARNs :

```
"Resource": [
         "arn:aws:ecs:us-east-1:123456789012:cluster/my-cluster-1",
         "arn:aws:ecs:us-east-1:123456789012:cluster/my-cluster-2"
```

Pour spécifier tous les clusters qui appartiennent à un compte spécifique, utilisez le caractère générique (*) :

```
"Resource": "arn:aws:ecs:us-east-1:123456789012:cluster/*"
```

Pour les définitions de tâches, vous pouvez spécifier la dernière révision ou une révision spécifique.

Pour spécifier toutes les révisions de la définition de tâche, utilisez le caractère générique (*) :

```
"Resource:arn:${Partition}:ecs:${Region}:${Account}:task-definition/
${TaskDefinitionFamilyName}:*"
```

Pour spécifier une révision de définition de tâche spécifique, utilisez ${TaskDefinitionRevisionNumber} :

```
"Resource:arn:${Partition}:ecs:${Region}:${Account}:task-definition/
${TaskDefinitionFamilyName}:${TaskDefinitionRevisionNumber}"
```

Pour voir des exemples de politiques Amazon ECS basées sur l'identité, consultez Exemples de politiques basées sur l'identité pour Amazon Elastic Container Service.

Clés de condition de politique pour Amazon ECS

Prend en charge les clés de condition de politique spécifiques au service : oui

Les administrateurs peuvent utiliser les politiques AWS JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Condition` (ou le bloc `Condition`) vous permet de spécifier des conditions lorsqu'une instruction est appliquée. L'élément `Condition` est facultatif. Vous pouvez créer des expressions conditionnelles qui utilisent des opérateurs de condition, tels que les signes égal ou inférieur à, pour faire correspondre la condition de la politique aux valeurs de la demande.

Si vous spécifiez plusieurs éléments `Condition` dans une instruction, ou plusieurs clés dans un seul élément `Condition`, AWS les évalue à l'aide d'une opération `AND` logique. Si vous spécifiez plusieurs valeurs pour une seule clé de condition, AWS évalue la condition à l'aide d'une `OR` opération logique. Toutes les conditions doivent être remplies avant que les autorisations associées à l'instruction ne soient accordées.

Vous pouvez aussi utiliser des variables d'espace réservé quand vous spécifiez des conditions. Par exemple, vous pouvez accorder à un utilisateur IAM l'autorisation d'accéder à une ressource uniquement si elle est balisée avec son nom d'utilisateur IAM. Pour plus d'informations, consultez Éléments d'une politique IAM : variables et identifications dans le Guide de l'utilisateur IAM.

AWS prend en charge les clés de condition globales et les clés de condition spécifiques au service. Pour voir toutes les clés de condition AWS globales, voir les clés de contexte de condition AWS globales dans le guide de l'utilisateur IAM.

Amazon ECS prend en charge les clés de condition spécifiques au service suivantes que vous pouvez utiliser pour fournir un filtrage précis pour vos politiques IAM :

Clé de condition	Description	Types d'évaluation
était : RequestTag /$ {} TagKey	La clé de contexte est formatée à l'"`aws:RequestTag/` *`tag-key`*`":"`*`tag-value`* "endroit où se *`tag-key`* *`tag-value`* trouvent une clé de balise et une paire de valeurs. Vérifie que la paire clé-valeur du tag est présente dans une AWS demande. Par exemple, vous pouvez vérifier que la requête comprend la clé d'étiquette `"Dept"` et qu'elle a la valeur `"Accounting"` .	Chaîne
était : ResourceTag /$ {} TagKey	La clé de contexte est formatée à l'"`aws:ResourceTag/` *`tag-key`*`":"`*`tag-value`* "endroit où se *`tag-key`* *`tag-value`* trouvent une clé de balise et une paire de valeurs. Vérifie que l'étiquette attachée à la ressource d'identité (utilisateur ou rôle) correspond aux nom et valeur de la clé spécifiée.	Chaîne
lois : TagKeys	Cette clé de contexte est formatée `"aws:TagKeys":"` *`tag-key`*" avec *`tag-key`* une liste de clés de balise sans valeurs (par exemple,`["Dept","Cost-Center"]`). Vérifie les clés de balise présentes dans une AWS demande.	Chaîne
sec : ResourceTag /$ {} TagKey	La clé de contexte est formatée à l'"`ecs:ResourceTag/` *`tag-key`*`":"`*`tag-value`* "endroit où se *`tag-key`* *`tag-value`* trouvent une clé de balise et une paire de valeurs.	Chaîne

Clé de condition	Description	Types d'évaluation
	Vérifie que l'étiquette attachée à la ressource d'identité (utilisateur ou rôle) correspond aux nom et valeur de la clé spécifiée.	
ecs:account-setting	La clé de contexte est formatée à l'"`ecs:account-setting":"` *`account-setting`* "endroit où se *`account-setting`* trouve le nom du paramètre du compte.	Chaîne
ecs : auto-assign-public-ip	La clé de contexte est formatée "`ecs:auto-assign-public-ip":"` *`value`*" avec *`value-`* « `true` » ou « `false` ».	Chaîne
ecs:capacité/fournisseur	La clé de contexte est formatée "`ecs:capacity-provider":"` *`capacity-provider-arn`* " là où se *`capacity-provider-arn`* trouve l'ARN du fournisseur de capacité.	ARN, Null
ecs:cluster	La clé de contexte est formatée "`ecs:cluster":"` *`cluster-arn`* " là où se *`cluster-arn`* trouve l'ARN du cluster Amazon ECS.	ARN, Null
ecs:compatibilité informatique	La clé de contexte est "`ecs:compute-compatability":"` *`compatability-type`* " formatée *`compatability-type`* avec `FARGATE` « », « `EC2` » ou « `EXTERNAL` ».	Chaîne
ecs:container-instances	La clé de contexte est formatée "`ecs:container-instances":"` *`container-instance-arns`* " là où *`container-instance-arns`* se trouvent une ou plusieurs instances ARNs de conteneur.	ARN, Null

Clé de condition	Description	Types d'évaluation
ecs:container-name	La clé de contexte est "`ecs:container-name":"` *`container-name`* " formatée *`container-instance-`* avec le nom d'un conteneur Amazon ECS défini dans la définition de tâche.	Chaîne
ecs : enable-execute-command	La clé de contexte est formatée "`ecs:enable-execute-command":"` *`value`*" avec *`value-`* « `true` » ou « `false` ».	Chaîne
ecs : enable-service-connect	La clé de contexte est formatée à l'"`ecs:enable-service-connect":"` *`value`*"endroit où se *`value`* trouve "`true`" ou"`false`".	Chaîne
ecs : enable-ebs-volumes	La clé de contexte est formatée à l'"`ecs:enable-ebs-volumes":"` *`value`*"endroit où se *`value`* trouve "`true`" ou"`false`".	Chaîne
ecs : enable-managed-tags	La clé de contexte est formatée à l'"`ecs:enable-managed-tags":"` *`value`*"endroit où se *`value`* trouve "`true`" ou"`false`".	Chaîne
ecs : enable-vpc-lattice	La clé de contexte est formatée à l'"`ecs:vpc-lattice":"` *`value`*"endroit où se *`value`* trouve "`true`" ou"`false`".	Chaîne
ecs : fargate-ephemeral-storage-kms -key	La clé de contexte est formatée "`ecs:fargate-ephemeral-storage-kms-key":"` *`key`*" là où se *`key`* trouve l'ID de la AWS KMS clé.	Chaîne
ecs:namespace	La clé de contexte est formatée "`ecs:namespace":"` *`namespace-arn`* " là où se *`namespace-arn`* trouve l'ARN de l'espace de AWS Cloud Map noms.	ARN, Null

Clé de condition	Description	Types d'évaluation
ecs:propagate-tags	La clé de contexte est formatée à l'`"ecs:propagate-tags":" value"`endroit où *value* est `"TASK_DEFINITION" "SERVICE"` , ou`"NONE"`.	Chaîne
ecs:service	La clé de contexte est formatée `"ecs:service":" service-arn "` là où se *service-arn* trouve l'ARN du service Amazon ECS.	ARN, Null
ecs:task-definition	La clé de contexte est formatée `"ecs:task-definition":" task-definition-arn "` là où se *task-definition-arn* trouve l'ARN pour la définition de tâche Amazon ECS.	ARN, Null
ecs:sous-réseau	La clé de contexte est formatée `"ecs:subnet":" subnets"` là où se *subnets* trouvent les sous-réseaux d'une tâche ou d'un service utilisant `awsvpc` le mode.	Chaîne
ecs:tâche	La clé de contexte est formatée `"ecs:task":" task-arn"` là où se *task-arn* trouve l'ARN du service Amazon ECS.	ARN, Null
ecs : task-cpu	La clé de contexte est formatée à l'`"ecs:task-cpu":"task-cpu"`endroit où se *task-cpu* trouve le processeur de tâches, sous la forme d'un entier avec 1024 = 1 vCPU.	Entier
ecs : mémoire de tâches	La clé de contexte est formatée `"ecs:task-memory":" task-memory "` là où se *task-memory* trouve la mémoire des tâches en MiB.	Entier

Pour voir la liste des clés de condition Amazon ECS, consultez Clés de condition pour Amazon Elastic Container Service dans la Référence de l'autorisation de service. Pour savoir avec quelles

actions et ressources vous pouvez utiliser une clé de condition, consultez Actions définies par Amazon Elastic Container Service.

Pour voir des exemples de politiques Amazon ECS basées sur l'identité, consultez Exemples de politiques basées sur l'identité pour Amazon Elastic Container Service.

Listes de contrôle d'accès (ACLs) dans Amazon ECS

Supports ACLs : Non

Les listes de contrôle d'accès (ACLs) contrôlent les principaux (membres du compte, utilisateurs ou rôles) autorisés à accéder à une ressource. ACLs sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Contrôle d'accès basé sur les attributs (ABAC) avec Amazon ECS

> **Important**
>
> Amazon ECS prend en charge le contrôle d'accès basé sur les attributs pour toutes les ressources Amazon ECS. Pour déterminer si vous pouvez utiliser des attributs pour définir le périmètre d'une action, utilisez le tableau Actions définies par Amazon ECS dans la Référence de l'autorisation de service. Vérifiez d'abord qu'il existe une ressource dans la colonne Ressource. Utilisez ensuite la colonne Clés de condition pour voir les clés de la action/resource combinaison.

Prise en charge d’ABAC (balises dans les politiques) : Oui

Le contrôle d’accès par attributs (ABAC) est une stratégie d’autorisation qui définit des autorisations en fonction des attributs. Dans AWS, ces attributs sont appelés balises. Vous pouvez associer des balises aux entités IAM (utilisateurs ou rôles) et à de nombreuses AWS ressources. L’étiquetage des entités et des ressources est la première étape d’ABAC. Vous concevez ensuite des politiques ABAC pour autoriser des opérations quand l’identification du principal correspond à celle de la ressource à laquelle il tente d’accéder.

L’ABAC est utile dans les environnements qui connaissent une croissance rapide et pour les cas où la gestion des politiques devient fastidieuse.

Pour contrôler l'accès basé sur des étiquettes, vous devez fournir les informations d'étiquette dans l'élément de condition d'une politique utilisant les clés de condition `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` ou `aws:TagKeys`.

Si un service prend en charge les trois clés de condition pour tous les types de ressources, alors la valeur pour ce service est Oui. Si un service prend en charge les trois clés de condition pour certains types de ressources uniquement, la valeur est Partielle.

Pour plus d'informations sur ABAC, consultez Définition d'autorisations avec l'autorisation ABAC dans le Guide de l'utilisateur IAM. Pour accéder à un didacticiel décrivant les étapes de configuration de l'ABAC, consultez Utilisation du contrôle d'accès par attributs (ABAC) dans le Guide de l'utilisateur IAM.

Pour en savoir plus sur l'identification des ressources Amazon ECS, consultez Balisage des ressources Amazon ECS.

Pour visualiser un exemple de politique basée sur l'identité permettant de limiter l'accès à une ressource en fonction des étiquettes de cette ressource, consultez Description des services Amazon ECS basée sur des balises.

Utilisation d'informations d'identification temporaires avec Amazon ECS

Prend en charge les informations d'identification temporaires : oui

Certains Services AWS ne fonctionnent pas lorsque vous vous connectez à l'aide d'informations d'identification temporaires. Pour plus d'informations, y compris celles qui Services AWS fonctionnent avec des informations d'identification temporaires, consultez Services AWS la section relative à l'utilisation d'IAM dans le guide de l'utilisateur d'IAM.

Vous utilisez des informations d'identification temporaires si vous vous connectez à l' AWS Management Console aide d'une méthode autre qu'un nom d'utilisateur et un mot de passe. Par exemple, lorsque vous accédez à AWS l'aide du lien d'authentification unique (SSO) de votre entreprise, ce processus crée automatiquement des informations d'identification temporaires. Vous créez également automatiquement des informations d'identification temporaires lorsque vous vous connectez à la console en tant qu'utilisateur, puis changez de rôle. Pour plus d'informations sur le changement de rôle, consultez Passage d'un rôle utilisateur à un rôle IAM (console) dans le Guide de l'utilisateur IAM.

Vous pouvez créer manuellement des informations d'identification temporaires à l'aide de l' AWS API AWS CLI or. Vous pouvez ensuite utiliser ces informations d'identification temporaires pour y accéder

AWS. AWS recommande de générer dynamiquement des informations d'identification temporaires au lieu d'utiliser des clés d'accès à long terme. Pour plus d'informations, consultez Informations d'identification de sécurité temporaires dans IAM.

Transférer les sessions d'accès pour Amazon ECS

Prend en charge les sessions d'accès direct (FAS) : oui

Lorsque vous utilisez un utilisateur ou un rôle IAM pour effectuer des actions AWS, vous êtes considéré comme un mandant. Lorsque vous utilisez certains services, vous pouvez effectuer une action qui initie une autre action dans un autre service. FAS utilise les autorisations du principal appelant et Service AWS, associées Service AWS à la demande, pour adresser des demandes aux services en aval. Les demandes FAS ne sont effectuées que lorsqu'un service reçoit une demande qui nécessite des interactions avec d'autres personnes Services AWS ou des ressources pour être traitée. Dans ce cas, vous devez disposer d'autorisations nécessaires pour effectuer les deux actions. Pour plus de détails sur une politique lors de la formulation de demandes FAS, consultez Transmission des sessions d'accès.

Fonctions du service pour Amazon ECS

Prend en charge les rôles de service : oui

Un rôle de service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer un rôle de service à partir d'IAM. Pour plus d'informations, consultez Création d'un rôle pour la délégation d'autorisations à un Service AWS dans le Guide de l'utilisateur IAM.

> **Warning**
>
> La modification des autorisations d'une fonction de service peut altérer la fonctionnalité d'Amazon ECS. Ne modifiez des rôles de service que quand Amazon ECS vous le conseille.

Rôles liés à un service pour Amazon ECS

Prend en charge les rôles liés aux services : Oui

Un rôle lié à un service est un type de rôle de service lié à un. Service AWS Le service peut endosser le rôle afin d'effectuer une action en votre nom. Les rôles liés à un service apparaissent dans votre

Compte AWS répertoire et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Pour plus d'informations sur la création ou la gestion des rôles liés à un service Amazon ECS service, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS.

Exemples de politiques basées sur l'identité pour Amazon Elastic Container Service

Par défaut, les utilisateurs et les rôles ne sont pas autorisés à créer ou à modifier les ressources Amazon ECS. Ils ne peuvent pas non plus effectuer de tâches à l'aide de l'API AWS Management Console, AWS Command Line Interface (AWS CLI) ou de AWS l'API. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM. L'administrateur peut ensuite ajouter les politiques IAM aux rôles et les utilisateurs peuvent assumer les rôles.

Pour apprendre à créer une politique basée sur l'identité IAM à l'aide de ces exemples de documents de politique JSON, consultez Création de politiques IAM (console) dans le Guide de l'utilisateur IAM.

Pour plus de détails sur les actions et les types de ressources définis par Amazon ECS, y compris le ARNs format de chaque type de ressource, consultez la section Actions, ressources et clés de condition pour Amazon Elastic Container Service dans la référence d'autorisation du service.

Rubriques

- Bonnes pratiques en matière de politique Amazon ECS
- Autoriser les utilisateurs d'Amazon ECS à consulter leurs propres autorisations
- Exemples de clusters Amazon ECS
- Exemples d'instances de conteneur Amazon ECS
- Exemples de définition de tâches Amazon ECS
- Exemple de tâche d'exécution d'Amazon ECS
- Exemple de tâche de démarrage d'Amazon ECS
- Répertorier et décrire des exemples de tâches Amazon ECS
- Créer un exemple de service Amazon ECS
- Exemple de mise à jour du service Amazon ECS
- Description des services Amazon ECS basée sur des balises
- Exemple de dérogation à l'espace de noms Refuser Amazon ECS Service Connect

Bonnes pratiques en matière de politique Amazon ECS

Les stratégies basées sur l'identité déterminent si une personne peut créer, consulter ou supprimer des ressources Amazon ECS dans votre compte. Ces actions peuvent entraîner des frais pour votre Compte AWS. Lorsque vous créez ou modifiez des politiques basées sur l'identité, suivez ces instructions et recommandations :

- Commencez AWS par les politiques gérées et passez aux autorisations du moindre privilège : pour commencer à accorder des autorisations à vos utilisateurs et à vos charges de travail, utilisez les politiques AWS gérées qui accordent des autorisations pour de nombreux cas d'utilisation courants. Ils sont disponibles dans votre Compte AWS. Nous vous recommandons de réduire davantage les autorisations en définissant des politiques gérées par les AWS clients spécifiques à vos cas d'utilisation. Pour plus d'informations, consultez politiques gérées par AWS ou politiques gérées par AWS pour les activités professionnelles dans le Guide de l'utilisateur IAM.
- Accordez les autorisations de moindre privilège : lorsque vous définissez des autorisations avec des politiques IAM, accordez uniquement les autorisations nécessaires à l'exécution d'une seule tâche. Pour ce faire, vous définissez les actions qui peuvent être entreprises sur des ressources spécifiques dans des conditions spécifiques, également appelées autorisations de moindre privilège. Pour plus d'informations sur l'utilisation d'IAM pour appliquer des autorisations, consultez politiques et autorisations dans IAM dans le Guide de l'utilisateur IAM.
- Utilisez des conditions dans les politiques IAM pour restreindre davantage l'accès : vous pouvez ajouter une condition à vos politiques afin de limiter l'accès aux actions et aux ressources. Par exemple, vous pouvez écrire une condition de politique pour spécifier que toutes les demandes doivent être envoyées via SSL. Vous pouvez également utiliser des conditions pour accorder l'accès aux actions de service si elles sont utilisées par le biais d'un service spécifique Service AWS, tel que AWS CloudFormation. Pour plus d'informations, consultez Conditions pour éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.
- Utilisez l'Analyseur d'accès IAM pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles : l'Analyseur d'accès IAM valide les politiques nouvelles et existantes de manière à ce que les politiques IAM respectent le langage de politique IAM (JSON) et les bonnes pratiques IAM. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées et fonctionnelles. Pour plus d'informations, consultez Validation de politiques avec IAM Access Analyzer dans le Guide de l'utilisateur IAM.
- Exiger l'authentification multifactorielle (MFA) : si vous avez un scénario qui nécessite des utilisateurs IAM ou un utilisateur root, activez l'authentification MFA pour une sécurité accrue.

Compte AWS Pour exiger la MFA lorsque des opérations d'API sont appelées, ajoutez des conditions MFA à vos politiques. Pour plus d'informations, consultez Sécurisation de l'accès aux API avec MFA dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur les bonnes pratiques dans IAM, consultez Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM.

Autoriser les utilisateurs d'Amazon ECS à consulter leurs propres autorisations

Cet exemple montre comment créer une politique qui permet aux utilisateurs IAM d'afficher les politiques en ligne et gérées attachées à leur identité d'utilisateur. Cette politique inclut les autorisations permettant d'effectuer cette action sur la console ou par programmation à l'aide de l'API AWS CLI or AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

Exemples de clusters Amazon ECS

La stratégie IAM suivante accorde l'autorisation de créer et de répertorier des clusters. Les actions `CreateCluster` et `ListClusters` n'acceptent aucune ressource. La définition de ressource est donc définie sur `*` pour toutes les ressources.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:CreateCluster",
                "ecs:ListClusters"
            ],
            "Resource": ["*"]
        }
    ]
}
```

La stratégie IAM suivante accorde l'autorisation de décrire et de supprimer un cluster spécifique. Les `DeleteCluster` actions `DescribeClusters` et acceptent les clusters ARNs en tant que ressources.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:DescribeClusters",
                "ecs:DeleteCluster"
            ],
            "Resource": ["arn:aws:ecs:us-east-1:<aws_account_id>:cluster/
<cluster_name>"]
        }
    ]
```

```
}
```

La stratégie IAM suivante peut être liée à un utilisateur ou à groupe, ce qui autoriserait cet utilisateur ou ce groupe à effectuer des opérations uniquement sur un cluster spécifique.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ecs:Describe*",
                "ecs:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "ecs:DeleteCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:ListContainerInstances",
                "ecs:RegisterContainerInstance",
                "ecs:SubmitContainerStateChange",
                "ecs:SubmitTaskStateChange"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ecs:us-east-1:<aws_account_id>:cluster/default"
        },
        {
            "Action": [
                "ecs:DescribeContainerInstances",
                "ecs:DescribeTasks",
                "ecs:ListTasks",
                "ecs:UpdateContainerAgent",
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "ArnEquals": {"ecs:cluster": "arn:aws:ecs:us-
east-1:<aws_account_id>:cluster/default"}
```

```
            }
        }
    ]
}
```

Exemples d'instances de conteneur Amazon ECS

L'enregistrement des instances de conteneur est géré par l'agent Amazon ECS, mais vous pouvez parfois avoir besoin d'autoriser un utilisateur à annuler manuellement l'enregistrement d'une instance d'un cluster. Il est possible que l'instance de conteneur ait été enregistrée accidentellement dans le mauvais cluster ou que l'instance ait été résiliée alors que des tâches étaient en cours d'exécution.

La stratégie IAM suivante permet à un utilisateur de répertorier et d'annuler l'enregistrement des instances de conteneur qui se trouvent dans un cluster donné :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:DeregisterContainerInstance",
                "ecs:ListContainerInstances"
            ],
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>"]
        }
    ]
}
```

La stratégie IAM qui suit permet à un utilisateur de décrire une instance de conteneur spécifiée dans un cluster donné. Pour accorder cette autorisation à toutes les instances de conteneur d'un cluster, vous pouvez remplacer l'UUID de l'instance de conteneur par `*`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:DescribeContainerInstances"],
            "Condition": {
```

```
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:container-instance/
<cluster_name>/<container_instance_UUID>"]
        }
    ]
}
```

Exemples de définition de tâches Amazon ECS

Les politiques IAM de définition de tâche ne prennent pas en charge les autorisations au niveau des ressources, mais la politique IAM suivante autorise un utilisateur à enregistrer, répertorier et décrire des définitions de tâche :

Si vous utilisez la console, vous devez ajouter `CloudFormation: CreateStack` en tant qu'`Action`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:RegisterTaskDefinition",
                "ecs:ListTaskDefinitions",
                "ecs:DescribeTaskDefinition"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Exemple de tâche d'exécution d'Amazon ECS

Les ressources pour `RunTask` sont des définitions de tâches. Pour limiter les clusters sur lesquels un utilisateur peut exécuter des définitions de tâches, vous pouvez les spécifier dans le bloc `Condition`. Cette méthode présente l'avantage de ne pas avoir besoin de répertorier les définitions de tâche et les clusters de vos ressources pour accorder un accès approprié. Vous pouvez appliquer l'une ou l'autre, ou les deux.

La politique IAM suivante accorde l'autorisation d'exécuter toute révision d'une définition de tâche spécifique sur un cluster spécifique :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:RunTask"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:task-definition/
<task_family>:*"]
        }
    ]
}
```

Exemple de tâche de démarrage d'Amazon ECS

Les ressources pour `StartTask` sont des définitions de tâches. Pour limiter les clusters et les instances de conteneur sur lesquels un utilisateur peut démarrer des définitions de tâches, vous pouvez les spécifier dans le bloc `Condition`. Cette méthode présente l'avantage de ne pas avoir besoin de répertorier les définitions de tâche et les clusters de vos ressources pour accorder un accès approprié. Vous pouvez appliquer l'une ou l'autre, ou les deux.

La stratégie IAM suivante accorde l'autorisation de lancer la révision d'une définition de tâche spécifique sur un cluster spécifique et une instance de conteneur spécifique.

> **Note**
>
> Dans cet exemple, lorsque vous appelez l'`StartTask`API avec le SDK AWS CLI ou un autre AWS SDK, vous devez spécifier la révision de la définition de tâche afin que le `Resource` mappage corresponde.

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": ["ecs:StartTask"],
            "Condition": {
                "ArnEquals": {
                    "ecs:cluster": "arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>",
                    "ecs:container-instances":
 ["arn:aws:ecs:<region>:<aws_account_id>:container-instance/<cluster_name>/
<container_instance_UUID>"]
                }
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:task-definition/
<task_family>:*"]
        }
    ]
}
```

Répertorier et décrire des exemples de tâches Amazon ECS

La stratégie IAM suivante autorise un utilisateur à répertorier les tâches pour un cluster spécifié :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:ListTasks"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["*"]
        }
    ]
}
```

La stratégie IAM suivante permet à un utilisateur de décrire une tâche spécifiée dans un cluster spécifié :

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:DescribeTasks"],
            "Condition": {
                "ArnEquals": {"ecs:cluster":
 "arn:aws:ecs:<region>:<aws_account_id>:cluster/<cluster_name>"}
            },
            "Resource": ["arn:aws:ecs:<region>:<aws_account_id>:task/<cluster_name>/
<task_UUID>"]
        }
    ]
}
```

Créer un exemple de service Amazon ECS

La stratégie IAM suivante permet à un utilisateur de créer des services Amazon ECS dans la AWS Management Console :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:Describe*",
                "application-autoscaling:PutScalingPolicy",
                "application-autoscaling:RegisterScalableTarget",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "ecs:List*",
                "ecs:Describe*",
                "ecs:CreateService",
                "elasticloadbalancing:Describe*",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:ListAttachedRolePolicies",
                "iam:ListRoles",
                "iam:ListGroups",
                "iam:ListUsers"
            ],
            "Resource": ["*"]
```

```
        }
    ]
}
```

Exemple de mise à jour du service Amazon ECS

La stratégie IAM suivante permet à un utilisateur de créer des services Amazon ECS dans la AWS Management Console :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:Describe*",
                "application-autoscaling:PutScalingPolicy",
                "application-autoscaling:DeleteScalingPolicy",
                "application-autoscaling:RegisterScalableTarget",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "ecs:List*",
                "ecs:Describe*",
                "ecs:UpdateService",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:ListAttachedRolePolicies",
                "iam:ListRoles",
                "iam:ListGroups",
                "iam:ListUsers"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Description des services Amazon ECS basée sur des balises

Vous pouvez utiliser des conditions dans votre stratégie basée sur l'identité pour contrôler l'accès aux ressources Amazon ECS en fonction des balises. Cet exemple montre comment créer une politique qui autorise la description de vos services. Toutefois, l'autorisation est accordée uniquement si

l'étiquette de service `Owner` a la valeur du nom d'utilisateur de cet utilisateur. Cette politique accorde également les autorisations nécessaires pour réaliser cette action sur la console.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "DescribeServices",
            "Effect": "Allow",
            "Action": "ecs;:DescribeServices",
            "Resource": "*"
        },
        {
            "Sid": "ViewServiceIfOwner",
            "Effect": "Allow",
            "Action": "ecs:DescribeServices",
            "Resource": "arn:aws:ecs:*:*:service/*",
            "Condition": {
                "StringEquals": {"ecs:ResourceTag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

Vous pouvez rattacher cette politique aux utilisateurs IAM de votre compte. Si un utilisateur nommé `richard-roe` tente de décrire un service Amazon ECS service, le service doit être balisé `Owner=richard-roe` ou `owner=richard-roe`. Dans le cas contraire, l'utilisateur se voit refuser l'accès. La clé de condition d'étiquette `Owner` correspond à la fois à `Owner` et à `owner`, car les noms de clé de condition ne sont pas sensibles à la casse. Pour plus d'informations, veuillez consulter la rubrique Éléments de stratégie JSON IAM : Condition dans le Guide de l'utilisateur IAM.

Exemple de dérogation à l'espace de noms Refuser Amazon ECS Service Connect

La politique IAM suivante interdit à un utilisateur de remplacer l'espace de noms Service Connect par défaut dans une configuration de service. L'espace de noms par défaut est défini dans le cluster. Cependant, vous pouvez le remplacer dans une configuration de service. Par souci de cohérence, pensez à configurer tous vos nouveaux services pour qu'ils utilisent le même espace de noms. Utilisez les clés de contexte suivantes pour demander aux services d'utiliser un espace

de noms spécifique. Remplacez la `<region>`, le `<aws_account_id>`, le `<cluster_name>` et l'`<namespace_id>` par les vôtres dans l'exemple suivant :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:CreateService",
                "ecs:UpdateService"
            ],
            "Condition": {
                "ArnEquals": {
                    "ecs:cluster": "arn:aws:ecs:<region>:<aws_account_id>:cluster/
<cluster_name>",
                    "ecs:namespace":
 "arn:aws:servicediscovery:<region>:<aws_account_id>:namespace/<namespace_id>"
                }
            },
            "Resource": "*"
        }
    ]
}
```

AWS politiques gérées pour Amazon Elastic Container Service

Pour ajouter des autorisations aux utilisateurs, aux groupes et aux rôles, il est plus facile d'utiliser des politiques AWS gérées que de les rédiger vous-même. Il faut du temps et de l'expertise pour créer des politiques gérées par le client IAM qui ne fournissent à votre équipe que les autorisations dont elle a besoin. Pour démarrer rapidement, vous pouvez utiliser nos politiques AWS gérées. Ces politiques couvrent les cas d'utilisation courants et sont disponibles dans votre AWS compte. Pour plus d'informations sur les politiques AWS gérées, voir les politiques AWS gérées dans le guide de l'utilisateur IAM.

AWS les services maintiennent et mettent à jour les politiques AWS gérées. Vous ne pouvez pas modifier les autorisations dans les politiques AWS gérées. Les services ajoutent parfois des autorisations supplémentaires à une politique AWS gérée pour prendre en charge de nouvelles fonctionnalités. Ce type de mise à jour affecte toutes les identités (utilisateurs, groupes et rôles)

auxquelles la politique est attachée. Les services sont plus susceptibles de mettre à jour une politique AWS gérée lorsqu'une nouvelle fonctionnalité est lancée ou lorsque de nouvelles opérations sont disponibles. Les services ne suppriment pas les autorisations d'une politique AWS gérée. Les mises à jour des politiques n'endommageront donc pas vos autorisations existantes.

En outre, AWS prend en charge les politiques gérées pour les fonctions professionnelles qui couvrent plusieurs services. Par exemple, la politique ReadOnlyAccess AWS gérée fournit un accès en lecture seule à tous les AWS services et ressources. Lorsqu'un service lance une nouvelle fonctionnalité, il AWS ajoute des autorisations en lecture seule pour les nouvelles opérations et ressources. Pour obtenir la liste des politiques de fonctions professionnelles et leurs descriptions, consultez la page politiques gérées par AWS pour les fonctions de tâche dans le Guide de l'utilisateur IAM.

Amazon ECS et Amazon ECR fournissent plusieurs politiques gérées et relations de confiance que vous pouvez associer aux utilisateurs, aux groupes, aux rôles, aux EC2 instances Amazon et aux tâches Amazon ECS qui permettent différents niveaux de contrôle sur les ressources et les opérations d'API. Vous pouvez appliquer ces politiques directement ou les utiliser comme points de départ pour créer vos propres politiques. Pour plus d'informations sur les stratégies gérées par Amazon ECR, veuillez consulter Stratégies gérées par Amazon ECR.

Amazon ECS_ FullAccess

Vous pouvez associer la politique `AmazonECS_FullAccess` à vos identités IAM. Cette politique accorde un accès administratif aux ressources Amazon ECS et accorde à une identité IAM (telle qu'un utilisateur, un groupe ou un rôle) un accès aux AWS services auxquels Amazon ECS est intégré afin d'utiliser toutes les fonctionnalités d'Amazon ECS. L'utilisation de cette stratégie permet d'accéder à toutes les fonctions Amazon ECS disponibles dans la AWS Management Console.

Pour consulter les autorisations associées à cette politique, consultez AmazonECS_ FullAccess dans le manuel AWS Managed Policy Reference.

Amazon ECSInfrastructure RolePolicyForVolumes

Vous pouvez associer la politique `AmazonECSInfrastructureRolePolicyForVolumes` gérée à vos entités IAM.

La politique accorde les autorisations nécessaires à Amazon ECS pour effectuer des appels AWS d'API en votre nom. Vous pouvez associer cette politique au rôle IAM que vous fournissez avec la configuration de votre volume lorsque vous lancez des tâches et des services Amazon ECS. Ce rôle permet à Amazon ECS de gérer les volumes associés à vos tâches. Pour plus d'informations, consultez le rôle IAM de l'infrastructure Amazon ECS.

Pour consulter les autorisations associées à cette politique, consultez Amazon ECSInfrastructure RolePolicyForVolumes dans le AWS Managed Policy Reference.

EC2ContainerServiceforEC2Rôle Amazon

Vous pouvez associer la politique `AmazonEC2ContainerServiceforEC2Role` à vos identités IAM. Cette politique accorde des autorisations administratives qui permettent aux instances de conteneur Amazon ECS de passer des appels AWS en votre nom. Pour de plus amples informations, veuillez consulter Rôle IAM d'instance de conteneur Amazon ECS.

Amazon ECS associe cette politique à un rôle de service qui permet à Amazon ECS d'effectuer des actions en votre nom contre des EC2 instances Amazon ou des instances externes.

Pour consulter les autorisations associées à cette politique, consultez Amazon EC2 ContainerServicefor EC2 Role dans le AWS Managed Policy Reference.

Considérations

Notez également les points suivants lorsque vous utilisez une politique IAM gérée par `AmazonEC2ContainerServiceforEC2Role`.

- En suivant les conseils de sécurité standard pour accorder le moindre privilège, vous pouvez modifier la stratégie IAM gérée par `AmazonEC2ContainerServiceforEC2Role` pour répondre à vos besoins spécifiques. Si l'une des autorisations accordées dans la stratégie gérée n'est pas nécessaire pour votre cas d'utilisation, créez une stratégie personnalisée et ajoutez uniquement les autorisations dont vous avez besoin. Par exemple, l'autorisation `UpdateContainerInstancesState` est fournie pour le drainage d'instances Spot. Si cette autorisation n'est pas nécessaire pour votre cas d'utilisation, excluez-la à l'aide d'une stratégie personnalisée.
- Les conteneurs qui s'exécutent sur vos instances de conteneur ont accès à toutes les autorisations fournies au rôle d'instance de conteneur par le biais des métadonnées d'instance. Nous vous recommandons de limiter les autorisations dans votre rôle d'instance de conteneur à la liste minimale d'autorisations fournie dans la stratégie gérée par `AmazonEC2ContainerServiceforEC2Role` présentée ci-dessous. Si les conteneurs dans vos tâches ont besoin d'autorisations supplémentaires qui ne sont pas répertoriées ici, nous vous recommandons de fournir leurs propres rôles IAM à ces tâches. Pour de plus amples informations, veuillez consulter Rôle IAM de la tâche Amazon ECS.

  Vous pouvez empêcher les conteneurs du bridge `docker0` d'accéder aux autorisations fournies au rôle de l'instance de conteneur, et ce tout en autorisant les autorisations fournies par Rôle

IAM de la tâche Amazon ECS en exécutant la commande iptables suivante sur vos instances de conteneur. Les conteneurs ne peuvent pas interroger les métadonnées d'instance lorsque cette règle est en vigueur. Notez que cette commande suppose une configuration bridge Docker par défaut et ne fonctionnera pas avec les conteneurs qui utilisent le mode réseau `host`. Pour de plus amples informations, veuillez consulter Mode réseau.

```
sudo yum install -y iptables-services; sudo iptables --insert DOCKER USER 1 --in-
interface docker+ --destination 169.254.169.254/32 --jump DROP
```

Pour garantir le maintien de la règle iptables après un redémarrage, vous devez l'enregistrer sur votre instance de conteneur. Pour l'AMI optimisée pour Amazon ECS utilisez la commande qui suit. Pour les autres systèmes d'exploitation, consultez la documentation correspondante.

- Pour l'AMI Amazon Linux 2 optimisée pour Amazon ECS :

  ```
  sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --
  now iptables
  ```

- Pour l'AMI Amazon Linux optimisée pour Amazon ECS :

  ```
  sudo service iptables save
  ```

Amazon EC2 ContainerServiceEventsRole

Vous pouvez associer la politique `AmazonEC2ContainerServiceEventsRole` à vos identités IAM. Cette politique accorde des autorisations qui permettent à Amazon EventBridge (anciennement CloudWatch Events) d'exécuter des tâches en votre nom. Cette stratégie peut être attachée au rôle IAM spécifié lorsque vous créez des tâches planifiées. Pour de plus amples informations, veuillez consulter Rôle EventBridge IAM d'Amazon ECS.

Pour consulter les autorisations associées à cette politique, consultez Amazon EC2 ContainerServiceEventsRole dans le AWS Managed Policy Reference.

Amazon ECSTask ExecutionRolePolicy

La politique IAM `AmazonECSTaskExecutionRolePolicy` gérée accorde les autorisations nécessaires à l'agent de conteneur Amazon ECS et aux agents de AWS Fargate conteneur pour effectuer des appels d' AWS API en votre nom. Cette stratégie peut être ajoutée à votre rôle IAM

d'exécution de tâche. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

Pour consulter les autorisations associées à cette politique, consultez Amazon ECSTask ExecutionRolePolicy dans le AWS Managed Policy Reference.

Amazon ECSService RolePolicy

La politique IAM gérée `AmazonECSServiceRolePolicy` permet à Amazon Elastic Container Service de gérer votre cluster. Cette politique peut être ajoutée à votre rôle lié au service AWSServiceRoleForECS.

Pour consulter les autorisations associées à cette politique, consultez Amazon ECSService RolePolicy dans le AWS Managed Policy Reference.

`AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecur`

Vous pouvez associer la politique `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` à vos entités IAM. Cette politique accorde un accès administratif à AWS Private Certificate Authority Secrets Manager et aux autres AWS services nécessaires pour gérer les fonctionnalités TLS d'Amazon ECS Service Connect en votre nom.

Pour consulter les autorisations associées à cette politique, consultez Amazon ECSInfrastructure RolePolicyForServiceConnectTransportLayerSecurity dans le AWS Managed Policy Reference.

`AWSApplicationAutoscalingECSServicePolicy`

Vous ne pouvez pas joindre de `AWSApplicationAutoscalingECSServicePolicy` à vos entités IAM. Cette stratégie est attachée à un rôle lié à un service qui permet à Application Auto Scaling d'effectuer des actions en votre nom. Pour plus d'informations, veuillez consulter Rôles liés à un service pour scalabilité automatique d'application.

Pour consulter les autorisations associées à cette politique, consultez la section Politique de AWSApplicationmise à l'échelle automatique ECSService dans le manuel AWS Managed Policy Reference.

`AWSCodeDeployRoleForECS`

Vous ne pouvez pas joindre de `AWSCodeDeployRoleForECS` à vos entités IAM. Cette politique est associée à un rôle lié à un service qui permet d' CodeDeploy effectuer des actions en votre nom.

Pour plus d'informations, consultez la section Créer un rôle de service pour CodeDeploy dans le Guide de AWS CodeDeploy l'utilisateur.

Pour consulter les autorisations associées à cette politique, consultez AWSCodeDeployRoleForECS dans le manuel AWS Managed Policy Reference.

`AWSCodeDeployRoleForECSLimited`

Vous ne pouvez pas joindre de `AWSCodeDeployRoleForECSLimited` à vos entités IAM. Cette politique est associée à un rôle lié à un service qui permet d' CodeDeploy effectuer des actions en votre nom. Pour plus d'informations, consultez la section Créer un rôle de service pour CodeDeploy dans le Guide de AWS CodeDeploy l'utilisateur.

Pour voir les autorisations de cette stratégie, consultez AWSCodeDeployRoleForECSLimited dans le AWS Guide de référence des stratégies gérées par.

`AmazonECSInfrastructureRolePolicyForLoadBalancers`

Vous pouvez associer la politique `AmazonECSInfrastructureRolePolicyForLoadBalancers` à vos entités IAM. Cette politique accorde des autorisations qui permettent à Amazon ECS de gérer les ressources Elastic Load Balancing en votre nom. La politique inclut :

- Autorisations en lecture seule pour décrire les auditeurs, les règles, les groupes cibles et l'état de santé des cibles
- Autorisations pour enregistrer et désenregistrer des cibles auprès de groupes cibles
- Autorisations permettant de modifier les écouteurs pour les équilibreurs de charge d'application et les équilibreurs de charge réseau
- Autorisations permettant de modifier les règles pour les équilibreurs de charge d'application

Ces autorisations permettent à Amazon ECS de gérer automatiquement les configurations de l'équilibreur de charge lors de la création ou de la mise à jour des services, garantissant ainsi le bon acheminement du trafic vers vos conteneurs.

Pour consulter les autorisations associées à cette politique, consultez Amazon ECSInfrastructure RolePolicyForLoadBalancers dans le AWS Managed Policy Reference.

`AmazonECSInfrastructureRolePolicyForVpcLattice`

Vous pouvez associer la politique `AmazonECSInfrastructureRolePolicyForVpcLattice` à vos entités IAM. Cette politique donne accès aux autres ressources de AWS service nécessaires pour gérer en votre nom la fonctionnalité VPC Lattice dans les charges de travail Amazon ECS.

Pour consulter les autorisations associées à cette politique, consultez Amazon ECSInfrastructure RolePolicyForVpcLattice dans le AWS Managed Policy Reference.

Permet d'accéder aux autres ressources AWS de service nécessaires pour gérer la fonctionnalité VPC Lattice dans les charges de travail Amazon ECS en votre nom.

Amazon ECS met à jour les politiques AWS gérées

Consultez les informations relatives aux mises à jour des politiques AWS gérées pour Amazon ECS depuis que ce service a commencé à suivre ces modifications. Pour recevoir des alertes automatiques sur les modifications apportées à cette page, abonnez-vous au flux RSS dans la page de l'historique des documents Amazon ECS.

Modification	Description	Date
Ajouter des autorisations à the section called “EC2ContainerServiceforEC2Rôle Amazon”	La politique IAM `AmazonEC2ContainerServiceforEC2Role` gérée a été mise à jour pour inclure l'`ecs:ListTagsForResource` autorisation. Cette autorisation permet à l'agent Amazon ECS de récupérer les balises d'instance de tâche et de conteneur via le point de terminaison des métadonnées des tâches (`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`).	4 août 2025

Modification	Description	Date
Ajoutez des autorisations à Amazon ECSInfrastructure RolePolicyForLoadBalancers.	La politique IAM `AmazonECS InfrastructureRole PolicyForLoadBalan cers` gérée a été mise à jour avec de nouvelles autorisat ions pour la description, le désenregistrement et l'enregis trement des groupes cibles.	25 juillet 2025
Ajouter une nouvelle the section called “`AmazonECS InfrastructureRole PolicyForLoadBalan cers`” politique	Ajout d'une nouvelle ECSInfrastructure RolePolic yForLoadBalancers politique Amazon qui fournit l'accès aux autres ressources de AWS service nécessaires pour gérer les équilibreurs de charge associés aux charges de travail Amazon ECS.	15 juillet 2025
Ajoutez des autorisations à Amazon ECSService RolePolicy.	La politique IAM `AmazonECS ServiceRolePolicy` gérée a été mise à jour avec de nouvelles AWS Cloud Map autorisations permettant à Amazon ECS de mettre à jour les attributs de AWS Cloud Map service pour les services gérés par Amazon ECS.	15 juillet 2025

Modification	Description	Date
Ajouter des autorisations à Amazon ECSService RolePolicy	La politique IAM `AmazonECS ServiceRolePolicy` gérée a été mise à jour avec de nouvelles AWS Cloud Map autorisations permettant à Amazon ECS de mettre à jour les attributs de AWS Cloud Map service pour les services gérés par Amazon ECS.	24 juin 2025
Ajouter des autorisations à Amazon ECSInfrastructure RolePolicyForVolumes	La `AmazonECSInfrastructureRolePolicyForVolumes` politique a été mise à jour pour ajouter l'`ec2:DescribeInstances` autorisation. Cette autorisation permet d'éviter les collisions de noms d'appareils pour les volumes Amazon EBS attachés à des tâches Amazon ECS exécutées sur la même instance de conteneur.	2 juin 2025
Ajouter un nouvel Amazon ECSInfrastructure RolePolicyForVpcLattice	Permet d'accéder aux autres ressources AWS de service nécessaires pour gérer la fonctionnalité VPC Lattice dans les charges de travail Amazon ECS en votre nom.	18 novembre 2024

Modification	Description	Date
Ajouter des autorisations à Amazon ECSInfrastructure RolePolicyForVolumes	La `AmazonECSInfrastructureRolePolicyForVolumes` politique a été mise à jour pour permettre aux clients de créer un volume Amazon EBS à partir d'un instantané.	10 octobre 2024
Autorisations ajoutées à the section called “Amazon ECS_FullAccess”	La `AmazonECS_FullAccess` politique a été mise à jour pour ajouter `iam:PassRole` des autorisations pour les rôles IAM pour un rôle nommé`ecsInfrastructureRole`. Il s'agit du rôle IAM par défaut créé par le AWS Management Console et destiné à être utilisé comme rôle d'infrastructure ECS permettant à Amazon ECS de gérer les volumes Amazon EBS attachés aux tâches ECS.	13 août 2024

Modification	Description	Date
Ajouter une nouvelle ECSInfrastructure RolePolicyForServiceConnectTransportLayerSecurity politique Amazon	Ajout d'une nouvelle ECSInfrastructure RolePolicyForServiceConnectTransportLayerSecurity politique Amazon qui fournit un accès administratif à AWS KMS Secrets Manager et permet aux fonctionnalités TLS d'Amazon ECS Service Connect de fonctionner correctement. AWS Private Certificate Authority	22 janvier 2024
Ajouter une nouvelle politique Amazon ECSInfrastructure RolePolicyForVolumes	La `AmazonECSInfrastructureRolePolicyForVolumes` politique a été ajoutée. La politique accorde les autorisations nécessaires à Amazon ECS pour effectuer des appels d' AWS API afin de gérer les volumes Amazon EBS associés aux charges de travail Amazon ECS.	11 janvier 2024
Ajouter des autorisations à Amazon ECSService RolePolicy	La politique IAM `AmazonECSServiceRolePolicy` gérée a été mise à jour avec de nouvelles `events` autorisations `autoscaling` et `autoscaling-plans` autorisations supplémentaires.	4 décembre 2023

Modification	Description	Date
Ajouter des autorisations à Amazon EC2 Container ServiceEventsRole	La politique IAM `AmazonECS ServiceRolePolicy` gérée a été mise à jour pour autoriser l'accès au fonctionn ement de l' AWS Cloud Map `DiscoverInstancesR evision` API.	4 octobre 2023
Ajouter des autorisations à Amazon EC2 Container Servicefor EC2 Role	La `AmazonEC2Container ServiceforEC2Role` politique a été modifiée pour ajouter l'`ecs:TagRe source` autorisation, qui inclut une condition qui limite l'autorisation uniquement aux clusters nouvellement créés et aux instances de conteneur enregistrées.	06 mars 2023
Ajouter des autorisations à the section called “Amazon ECS_ FullAccess”	La `AmazonECS_FullAcce ss` politique a été modifiée pour ajouter l'`elasticlo adbalancing:AddTag s` autorisation, qui inclut une condition qui limite l'autoris ation uniquement aux équilibre urs de charge, aux groupes cibles, aux règles et aux écouteurs créés récemment . Cette autorisation n'autoris e pas l'ajout de balises à des ressources Elastic Load Balancing déjà créées.	4 janvier 2023

Modification	Description	Date
Amazon ECS a commencé à assurer le suivi des modificat ions	Amazon ECS a commencé à suivre les modifications apportées AWS à ses politique s gérées.	8 juin 2021

Suppression progressive des politiques IAM AWS gérées pour Amazon Elastic Container Service

Les politiques IAM AWS gérées suivantes sont progressivement supprimées. Ces stratégies sont désormais remplacées par les stratégies mises à jour. Nous vous recommandons de mettre à jour vos utilisateurs ou rôles afin d'utiliser les politiques mises à jour.

Amazon EC2 ContainerServiceFullAccess

> **Important**
>
> La politique IAM gérée `AmazonEC2ContainerServiceFullAccess` sera progressivement abandonnée à compter du 29 janvier 2021, en réponse à la constatation d'un problème de sécurité lié à l'autorisation `iam:passRole`. Cette autorisation autorise l'accès à toutes les ressources, y compris les informations d'identification aux rôles du compte. La politique étant progressivement abandonnée, vous ne pouvez pas attacher la politique à de nouveaux utilisateurs ou rôles. Tous les utilisateurs ou rôles auxquels la stratégie est déjà attachée peuvent continuer à l'utiliser. Toutefois, nous vous recommandons de mettre à jour vos utilisateurs ou rôles afin d'utiliser la politique gérée par `AmazonECS_FullAccess` à la place. Pour de plus amples informations, veuillez consulter Migration vers la stratégie gérée `AmazonECS_FullAccess`.

Amazon EC2 ContainerServiceRole

> **Important**
>
> La politique IAM gérée `AmazonEC2ContainerServiceRole` sera progressivement abandonnée. Elle est désormais remplacée par le rôle lié au service Amazon ECS service.

Pour de plus amples informations, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS.

Amazon EC2 ContainerServiceAutoscaleRole

> **Important**
>
> La stratégie IAM gérée `AmazonEC2ContainerServiceAutoscaleRole` sera progressivement abandonnée. Elle est désormais remplacée par le rôle lié au service Application Auto Scaling pour Amazon ECS service. Pour plus d'informations, consultez Rôles liés aux services pour Application Auto Scaling dans le Guide de l'utilisateur Application Auto Scaling.

Migration vers la stratégie gérée **`AmazonECS_FullAccess`**

La stratégie IAM gérée `AmazonEC2ContainerServiceFullAccess` a été progressivement abandonnée à compter du 29 janvier 2021, en réponse à la constatation d'un problème de sécurité lié à l'autorisation `iam:passRole`. Cette autorisation autorise l'accès à toutes les ressources, y compris les informations d'identification aux rôles du compte. La politique étant progressivement abandonnée, vous ne pouvez pas attacher la politique à de nouveaux groupes, utilisateurs ou rôles. Tous les groupes, utilisateurs ou rôles auxquels la stratégie est déjà attachée peuvent continuer à l'utiliser. Toutefois, nous vous recommandons de mettre à jour vos groupes, utilisateurs ou rôles afin d'utiliser la politique gérée par `AmazonECS_FullAccess` à la place.

Les autorisations accordées par la politique `AmazonECS_FullAccess` incluent la liste complète des autorisations nécessaires pour utiliser ECS en tant qu'administrateur. Si vous utilisez actuellement des autorisations accordées par la `AmazonEC2ContainerServiceFullAccess` politique qui ne figurent pas dans la `AmazonECS_FullAccess` politique, vous pouvez les ajouter à une déclaration de politique intégrée. Pour de plus amples informations, veuillez consulter AWS politiques gérées pour Amazon Elastic Container Service.

Procédez comme suit pour déterminer si vous disposez de groupes, d'utilisateurs ou de rôles qui utilisent actuellement la politique IAM gérée par `AmazonEC2ContainerServiceFullAccess`. Ensuite, mettez-les à jour pour détacher la stratégie précédente et attachez la stratégie `AmazonECS_FullAccess`.

Pour mettre à jour un groupe, un utilisateur ou un rôle afin d'utiliser la politique AmazonECS_ FullAccess ()AWS Management Console

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation, choisissez Policies (Stratégies) et recherchez et sélectionnez la stratégie `AmazonEC2ContainerServiceFullAccess`.
3. Cliquez sur l'onglet Policy usage (Utilisation de la stratégie) qui affiche tout rôle IAM qui utilise actuellement cette stratégie.
4. Pour chaque rôle IAM qui utilise actuellement la `AmazonEC2ContainerServiceFullAccess` stratégie, sélectionnez le rôle et suivez les étapes suivantes pour détacher la stratégie progressivement supprimée et l'associer. `AmazonECS_FullAccess`

   a. Dans l'onglet Autorisations, choisissez le X à côté de la EC2 ContainerServiceFullAccess politique Amazon.

   b. Choisissez Add permissions (Ajouter des autorisations).

   c. Choisissez Joindre directement les politiques existantes, recherchez et sélectionnez la FullAccess politique AmazonECS_, puis choisissez Next : Review.

   d. Vérifiez vos modifications et choisissez Add permissions (Ajouter des autorisations).

   e. Répétez ces étapes pour chaque groupe, utilisateur ou rôle qui utilise la politique `AmazonEC2ContainerServiceFullAccess`.

Pour mettre à jour un groupe, un utilisateur ou un rôle afin d'utiliser la politique **`AmazonECS_FullAccess`** (AWS CLI)

1. Utilisez la generate-service-last-accessed-detailscommande pour générer un rapport contenant des informations détaillées sur la date à laquelle la politique d'élimination progressive a été utilisée pour la dernière fois.

   ```
   aws iam generate-service-last-accessed-details \
        --arn arn:aws:iam::aws:policy/AmazonEC2ContainerServiceFullAccess
   ```

   Exemple de sortie :

   ```
   {
       "JobId": "32bb1fb0-1ee0-b08e-3626-ae83EXAMPLE"
   }
   ```

2. Utilisez l'ID de tâche de la sortie précédente avec la get-service-last-accessed-detailscommande pour récupérer le dernier rapport consulté du service. Ce rapport affiche le nom de ressource Amazon (ARN) des entités IAM qui ont utilisé la politique de suppression progressive pour la dernière fois.

```
aws iam get-service-last-accessed-details \
      --job-id 32bb1fb0-1ee0-b08e-3626-ae83EXAMPLE
```

3. Utilisez l'une des commandes suivantes pour détacher la politique `AmazonEC2ContainerServiceFullAccess` d'un groupe, d'un utilisateur ou d'un rôle.

   - detach-group-policy
   - detach-role-policy
   - detach-user-policy

4. Utilisez l'une des commandes suivantes pour attacher la politique `AmazonECS_FullAccess` à un groupe, un utilisateur ou un rôle.

   - attach-group-policy
   - attach-role-policy
   - attach-user-policy

Utilisation des rôles liés à un service pour Amazon ECS

Amazon Elastic Container Service utilise des AWS Identity and Access Management rôles liés à un service (IAM). Un rôle lié à un service est un type unique de rôle IAM lié directement à Amazon ECS. Les rôles liés à un service sont prédéfinis par Amazon ECS et incluent toutes les autorisations requises par le service pour appeler d'autres services AWS en votre nom.

Un rôle lié à un service simplifie la configuration d'Amazon ECS, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Amazon ECS définit les autorisations de ses rôles liés à un service ; sauf définition contraire, seul Amazon ECS peut endosser ses rôles. Les autorisations définies comprennent la politique d’approbation et la politique d’autorisation. De plus, cette politique d’autorisation ne peut pas être attachée à une autre entité IAM.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez la section AWS Services qui fonctionnent avec IAM et recherchez les services dont la valeur est Oui dans la colonne Rôles liés à un service. Sélectionnez un Oui ayant un lien pour consulter la documentation du rôle lié à un service, pour ce service.

Autorisations du rôle lié à un service pour Amazon ECS

Amazon ECS utilise le rôle lié à un service nommé AWSService RoleFor ECS.

Le rôle lié au service AWSService RoleFor ECS fait confiance aux services suivants pour assumer le rôle :

- `ecs.amazonaws.com`

La politique d'autorisation des rôles nommée Amazon ECSService RolePolicy permet à Amazon ECS d'effectuer les actions suivantes sur les ressources spécifiées :

- Action : lorsque vous utilisez le mode réseau de `awsvpc` pour vos tâches Amazon ECS, Amazon ECS gère le cycle de vie des interfaces réseau élastiques associées à la tâche. Cela inclut également les balises que Amazon ECS ajoute à vos interfaces réseau élastiques.
- Action : lorsque vous utilisez un équilibreur de charge avec votre compte Amazon ECS, Amazon ECS gère l'enregistrement et l'annulation de l'enregistrement des ressources avec l'équilibreur de charge.
- Action : Lorsque vous utilisez Amazon ECS Service Discovery, Amazon ECS gère la Route 53 et les AWS Cloud Map ressources nécessaires au bon fonctionnement de la découverte de services.
- Action : lorsque vous utilisez le service de mise à l'échelle Amazon ECS, Amazon ECS gère les ressources Auto Scaling requises.
- Action : Amazon ECS crée et gère des CloudWatch alarmes et des flux de journaux qui facilitent la surveillance de vos ressources Amazon ECS.
- Action : lorsque vous utilisez Amazon ECS Exec, Amazon ECS gère les autorisations nécessaires pour démarrer des sessions Amazon ECS Exec pour vos tâches.
- Action : lorsque vous utilisez Amazon ECS Service Connect, Amazon ECS gère les ressources AWS Cloud Map requises pour utiliser la fonctionnalité.
- Action : Lorsque vous utilisez des fournisseurs de capacité Amazon ECS, Amazon ECS gère les autorisations requises pour modifier le groupe Auto Scaling et ses EC2 instances Amazon.
- Action : Amazon ECS peut mettre à jour les attributs de AWS Cloud Map service pour les services gérés par Amazon ECS.

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour plus d’informations, consultez Autorisations de rôles liés à un service dans le Guide de l’utilisateur IAM.

Création d'un rôle lié à un service pour Amazon ECS

Dans la plupart des cas, vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un cluster ou que vous créez ou mettez à jour un service dans l' AWS Management Console AWS CLI AWS API, Amazon ECS crée le rôle lié au service pour vous. Si le rôle AWSServiceRoleForECS n'apparaît pas après avoir créé un cluster, effectuez les opérations suivantes pour résoudre le problème :

- Vérifiez et configurez les autorisations de manière à permettre à Amazon ECS de créer, modifier ou supprimer un rôle lié à un service en votre nom. Pour plus d'informations, consultez Autorisations de rôles liés à un service dans le Guide de l'utilisateur IAM.
- Réessayez l'opération de création de cluster ou créez manuellement le rôle lié à un service.

  Vous pouvez utiliser la console IAM pour créer le rôle lié au service AWSServiceRoleForECS. Dans l'API AWS CLI ou dans l' AWS API, créez un rôle lié à un service avec le nom du `ecs.amazonaws.com` service. Pour plus d'informations, consultez Création d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

> **Important**
>
> Ce rôle lié à un service peut apparaître dans votre compte si vous avez effectué une action dans un autre service qui utilise les fonctions prises en charge par ce rôle.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un cluster, ou lorsque vous créez ou mettez à jour un service, Amazon ECS crée à nouveau le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service, vous pouvez utiliser le même processus IAM pour recréer le rôle.

Modification d'un rôle lié à un service pour Amazon ECS

Amazon ECS ne vous permet pas de modifier le rôle lié au service AWSService RoleFor ECS. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas changer le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description

du rôle à l'aide d'IAM. Pour plus d'informations, voir Mettre à jour un rôle lié à un service dans le Guide de l'utilisateur IAM.

Suppression d'un rôle lié à un service pour Amazon ECS

Si vous n'avez plus besoin d'utiliser une fonctionnalité ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer les ressources de votre rôle lié à un service avant de pouvoir les supprimer manuellement.

> **Note**
>
> Si le service Amazon ECS utilise le rôle lorsque vous essayez de supprimer les ressources, la suppression peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour vérifier si une session est active pour le rôle lié à un service

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation, choisissez Roles et choisissez le nom AWSService RoleFor ECS (pas la case à cocher).
3. Sur la page Résumé, choisissez Access Advisor et consultez l'activité récente pour le rôle lié à un service.

   > **Note**
   >
   > Si vous ne savez pas si Amazon ECS utilise le rôle AWSService RoleFor ECS, vous pouvez essayer de le supprimer. Si le service utilise le rôle, la suppression échoue et vous avez accès aux régions dans lesquelles le rôle est utilisé. Si le rôle est utilisé, vous devez attendre que la session se termine avant de pouvoir le supprimer. Vous ne pouvez pas révoquer la session d'un rôle lié à un service.

Pour supprimer les ressources Amazon ECS utilisées par le rôle AWSService RoleFor lié au service ECS

Vous devez supprimer tous les clusters Amazon ECS dans toutes les AWS régions avant de pouvoir supprimer le rôle AWSService RoleFor ECS.

1. Arrêtez tous les services Amazon ECS en indiquant 0 comme nombre souhaité dans toutes les régions, puis supprimez-les. Pour plus d'informations, consultez Mettre à jour un service Amazon ECS et Supprimer un service Amazon ECS à l'aide de la console.
2. Forcez l'annulation de l'inscription de toutes les instances de conteneur de tous les clusters dans toutes les régions. Pour de plus amples informations, veuillez consulter Annulation de l'enregistrement d'une instance de conteneur Amazon ECS.
3. Supprimez tous les clusters Amazon ECS dans toutes les régions. Pour de plus amples informations, veuillez consulter Suppression d'un cluster Amazon ECS.

Pour supprimer manuellement le rôle lié à un service à l'aide d'IAM

Utilisez la console IAM, le AWS CLI, ou l' AWS API pour supprimer le rôle lié au service AWSService RoleFor ECS. Pour plus d'informations, consultez Suppression d'un rôle lié à un service dans le Guide de l'utilisateur IAM.

Régions prises en charge pour les rôles liés à un service Amazon ECS

Amazon ECS prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour plus d'informations, consultez Régions et points de terminaison AWS.

Rôles IAM pour Amazon ECS

Un rôle IAM est une identité IAM que vous pouvez créer dans votre compte et qui dispose d'autorisations spécifiques. Dans Amazon ECS, vous pouvez créer des rôles pour accorder des autorisations à des ressources Amazon ECS telles que des conteneurs ou des services.

Les rôles requis par Amazon ECS dépendent de la définition de tâche, du type de lancement et des fonctionnalités que vous utilisez. Utilisez le tableau suivant pour déterminer les rôles IAM dont vous avez besoin pour Amazon ECS.

Rôle	Définition	En cas de besoin	En savoir plus
Rôle d'exécution de tâche	Ce rôle permet à Amazon ECS d'utiliser d'autres AWS services en votre nom.	Votre tâche est hébergée sur AWS Fargateou sur des instances externes et :	Rôle IAM d'exécution de tâche Amazon ECS

Rôle	Définition	En cas de besoin	En savoir plus
		• extrait une image de conteneur depuis un référenti el privé Amazon ECR. • extrait une image de conteneur d'un référentiel privé Amazon ECR dans un compte différent de celui qui exécute la tâche. • envoie les journaux des conteneurs à CloudWatch Logs à l'aide du pilote de `awslogs` journal. Votre tâche est hébergée sur l'une AWS Fargate ou l'autre EC2 instance Amazon et : • utilise l'authent ification du registre privé. • utilise la surveilla nce du temps d'exécution. • la définition de la tâche fait référence à des données sensibles à l'aide	

Rôle	Définition	En cas de besoin	En savoir plus
		des secrets de Secrets Manager ou AWS des paramètres du magasin de paramètres de Systems Manager.	
Rôle de tâche	Ce rôle permet au code de votre application (sur le conteneur) d'utiliser d'autres AWS services.	Votre application accède à d'autres AWS services, tels qu'Amazon S3.	Rôle IAM de la tâche Amazon ECS
Rôle de l'instance de conteneur	Ce rôle permet à vos EC2 instances ou à des instances externes de s'enregistrer auprès du cluster.	Votre tâche est hébergée sur des EC2 instances Amazon ou sur une instance externe.	Rôle IAM d'instance de conteneur Amazon ECS
Rôle dans Amazon ECS Anywhere	Ce rôle permet à vos instances externes d'y accéder AWS APIs.	Votre tâche est hébergée sur des instances externes.	Rôle IAM dans Amazon ECS Anywhere
Rôle de l'infrastructure Amazon ECS pour les équilibreurs de charge	Ce rôle permet à Amazon ECS de gérer les ressources de l'équilibreur de charge dans vos clusters en votre nom pour les blue/green déploiements.	Vous souhaitez utiliser les blue/green déploiements Amazon ECS.	Rôle IAM de l'infrastructure Amazon ECS pour les équilibreurs de charge

Rôle	Définition	En cas de besoin	En savoir plus
CodeDeploy Rôle Amazon ECS	Ce rôle permet CodeDeploy de mettre à jour vos services.	Vous utilisez le type de déploiement CodeDeploy bleu/vert pour déployer des services.	Rôle CodeDeploy IAM d'Amazon ECS
EventBridge Rôle Amazon ECS	Ce rôle permet EventBridge de mettre à jour vos services.	Vous utilisez les EventBridge règles et les objectifs pour planifier vos tâches.	Rôle EventBridge IAM d'Amazon ECS

Rôle	Définition	En cas de besoin	En savoir plus
Rôle de l'infrast ructure Amazon ECS	Ce rôle permet à Amazon ECS de gérer les ressources d'infrastructure de vos clusters.	• Vous souhaitez associer des volumes Amazon EBS à vos tâches Amazon ECS de type Fargate ou de EC2 lancement . Le rôle d'infrast ructure permet à Amazon ECS de gérer les volumes Amazon EBS pour vos tâches. • Vous souhaitez utiliser le protocole TLS (Transport Layer Security) pour chiffrer le trafic entre vos services Amazon ECS Service Connect. • Vous souhaitez créer des groupes cibles VPC Lattice.	Rôle IAM dans l'infrastructure Amazon ECS

Bonnes pratiques pour les rôles IAM dans Amazon ECS

Les rôles requis par Amazon ECS dépendent de la définition de tâche, du type de lancement et des fonctionnalités que vous utilisez. Nous vous recommandons de créer des rôles distincts dans le tableau au lieu de partager des rôles.

Rôle	Définition	En cas de besoin	En savoir plus
Rôle d'exécution de tâche	Ce rôle permet à Amazon ECS d'utiliser d'autres AWS services en votre nom.	Votre tâche est hébergée sur AWS Fargateou sur des instances externes et : • extrait une image de conteneur depuis un référentiel privé Amazon ECR. • extrait une image de conteneur d'un référentiel privé Amazon ECR dans un compte différent de celui qui exécute la tâche. • envoie les journaux des conteneurs à CloudWatch Logs à l'aide du pilote de `awslogs` journal. Votre tâche est hébergée sur l'une AWS Fargate ou l'autre EC2 instance Amazon et : • utilise l'authentification du registre privé.	Rôle IAM d'exécution de tâche Amazon ECS

Rôle	Définition	En cas de besoin	En savoir plus
		• utilise la surveillance du temps d'exécution. • la définition de la tâche fait référence à des données sensibles à l'aide des secrets de Secrets Manager ou AWS des paramètres du magasin de paramètres de Systems Manager.	
Rôle de tâche	Ce rôle permet au code de votre application (sur le conteneur) d'utiliser d'autres AWS services.	Votre application accède à d'autres AWS services, tels qu'Amazon S3.	Rôle IAM de la tâche Amazon ECS
Rôle de l'instance de conteneur	Ce rôle permet à vos EC2 instances ou à des instances externes de s'enregistrer auprès du cluster.	Votre tâche est hébergée sur des EC2 instances Amazon ou sur une instance externe.	Rôle IAM d'instance de conteneur Amazon ECS
Rôle dans Amazon ECS Anywhere	Ce rôle permet à vos instances externes d'y accéder AWS APIs.	Votre tâche est hébergée sur des instances externes.	Rôle IAM dans Amazon ECS Anywhere

Rôle	Définition	En cas de besoin	En savoir plus
CodeDeploy Rôle Amazon ECS	Ce rôle permet CodeDeploy de mettre à jour vos services.	Vous utilisez le type de déploiement CodeDeploy bleu/vert pour déployer des services.	Rôle CodeDeploy IAM d'Amazon ECS
EventBridge Rôle Amazon ECS	Ce rôle permet EventBridge de mettre à jour vos services.	Vous utilisez les EventBridge règles et les objectifs pour planifier vos tâches.	Rôle EventBridge IAM d'Amazon ECS

Rôle	Définition	En cas de besoin	En savoir plus
Rôle de l'infrastructure Amazon ECS	Ce rôle permet à Amazon ECS de gérer les ressources d'infrastructure de vos clusters.	• Vous souhaitez associer des volumes Amazon EBS à vos tâches Amazon ECS de type Fargate ou de EC2 lancement . Le rôle d'infrastructure permet à Amazon ECS de gérer les volumes Amazon EBS pour vos tâches. • Vous souhaitez utiliser le protocole TLS (Transport Layer Security) pour chiffrer le trafic entre vos services Amazon ECS Service Connect. • Vous souhaitez créer des groupes cibles VPC Lattice.	Rôle IAM dans l'infrastructure Amazon ECS

Rôle de tâche

Nous vous recommandons d'attribuer un rôle de tâche. Son rôle peut être distingué de celui de l' EC2 instance Amazon sur laquelle il s'exécute. L'attribution d'un rôle à chaque tâche est conforme au principe de l'accès au moindre privilège et permet un contrôle plus précis des actions et des ressources.

Lorsque vous ajoutez un rôle de tâche à une définition de tâche, l'agent de conteneur Amazon ECS crée automatiquement un jeton avec un ID d'informations d'identification unique (par exemple,

`12345678-90ab-cdef-1234-567890abcdef`) pour la tâche. Ce jeton et les informations d'identification du rôle sont ensuite ajoutés au cache interne de l'agent. L'agent remplit la variable d'environnement `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` du conteneur avec l'URI de l'ID d'informations d'identification (par exemple, `/v2/credentials/12345678-90ab-cdef-1234-567890abcdef`).

Vous pouvez récupérer manuellement les informations d'identification du rôle temporaire depuis un conteneur en ajoutant la variable d'environnement à l'adresse IP de l'agent de conteneur Amazon ECS et en exécutant la commande `curl` sur la chaîne résultante.

```
curl 169.254.170.2$AWS_CONTAINER_CREDENTIALS_RELATIVE_URI
```

La sortie attendue est la suivante :

```
{
 "RoleArn": "arn:aws:iam::123456789012:role/SSMTaskRole-SSMFargateTaskIAMRole-
DASWWSF2WGD6",
 "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
 "Token": "IQoJb3JpZ2luX2VjEEM/Example==",
 "Expiration": "2021-01-16T00:51:53Z"
}
```

Les nouvelles versions du AWS SDKs récupèrent automatiquement ces informations d'identification à partir de la variable d'`AWS_CONTAINER_CREDENTIALS_RELATIVE_URI`environnement lors des appels AWS d'API. Pour plus d'informations sur le renouvellement des informations d'identification, voir Renouveler les AWS informations d'identification sur RePost.

La sortie inclut une paire de clés d'accès composée d'un identifiant de clé d'accès secrète et d'une clé secrète que votre application utilise pour accéder AWS aux ressources. Il inclut également un jeton qui AWS permet de vérifier que les informations d'identification sont valides. Par défaut, les informations d'identification attribuées aux tâches utilisant des rôles de tâches sont valides pendant six heures. Ensuite, elles sont automatiquement soumises à une rotation par l'agent de conteneur Amazon ECS.

Rôle d'exécution de tâche

Le rôle d'exécution des tâches est utilisé pour accorder à l'agent de conteneur Amazon ECS l'autorisation d'appeler des actions d' AWS API spécifiques en votre nom. Par exemple, lorsque vous

l'utilisez AWS Fargate, Fargate a besoin d'un rôle IAM lui permettant d'extraire des images d'Amazon ECR et d'écrire des journaux dans Logs. CloudWatch Un rôle IAM est également requis lorsqu'une tâche fait référence à un secret stocké dans AWS Secrets Manager, tel qu'un secret d'extraction d'image.

> **Note**
>
> Si vous extrayez des images en tant qu'utilisateur authentifié, vous êtes moins susceptible d'être impacté par les modifications apportées aux limites de taux d'extraction de Docker Hub. Pour plus d'informations, veuillez consulter Authentification de registre privé pour les instances de conteneur (langue française non garantie).
> En utilisant Amazon ECR et Amazon ECR Public, vous pouvez éviter les limites imposées par Docker. Si vous extrayez des images depuis Amazon ECR, cela permet également de raccourcir les temps d'extraction du réseau et de réduire les modifications de transfert de données lorsque le trafic quitte votre VPC.

> **Important**
>
> Lorsque vous utilisez Fargate, vous devez vous authentifier auprès d'un registre d'images privé à l'aide de `repositoryCredentials`. Il n'est pas possible de définir les variables d'environnement de l'agent de conteneur Amazon ECS `ECS_ENGINE_AUTH_TYPE` ou `ECS_ENGINE_AUTH_DATA`, ou de modifier le fichier `ecs.config` pour les tâches hébergées sur Fargate. Pour plus d'informations, veuillez consulter Authentification de registre privé pour les tâches (langue française non garantie).

Rôle de l'instance de conteneur

La stratégie IAM gérée par `AmazonEC2ContainerServiceforEC2Role` doit inclure les autorisations suivantes. Conformément aux conseils de sécurité standard pour accorder le moindre privilège, la stratégie gérée par `AmazonEC2ContainerServiceforEC2Role` peut être utilisée comme guide. Si vous n'avez pas besoin des autorisations accordées dans la stratégie gérée pour votre cas d'utilisation, créez une stratégie personnalisée et ajoutez uniquement les autorisations dont vous avez besoin.

- `ec2:DescribeTags`— (Facultatif) Permet à un principal de décrire les balises associées à une EC2 instance Amazon. Cette autorisation est utilisée par l'agent de conteneur Amazon ECS pour

prendre en charge la propagation des balises de ressources. Pour de plus amples informations, veuillez consulter Comment les ressources sont étiquetées.

- `ecs:CreateCluster`— (Facultatif) Permet à un principal de créer un cluster Amazon ECS. Cette autorisation est utilisée par l'agent de conteneur Amazon ECS pour créer un cluster `default`, s'il n'existe pas déjà.
- `ecs:DeregisterContainerInstance`— (Facultatif) Permet à un principal de désenregistrer une instance de conteneur Amazon ECS d'un cluster. L'agent de conteneur Amazon ECS n'appelle pas cette opération d'API, mais cette autorisation est conservée pour garantir la rétrocompatibilité.
- `ecs:DiscoverPollEndpoint`— (Obligatoire) Cette action renvoie les points de terminaison que l'agent de conteneur Amazon ECS utilise pour demander des mises à jour.
- `ecs:Poll`— (Obligatoire) Permet à l'agent de conteneur Amazon ECS de communiquer avec le plan de contrôle Amazon ECS pour signaler les modifications de l'état des tâches.
- `ecs:RegisterContainerInstance`— (Obligatoire) Permet à un principal d'enregistrer une instance de conteneur auprès d'un cluster. Cette autorisation est utilisée par l'agent de conteneur Amazon ECS pour enregistrer l' EC2 instance Amazon auprès d'un cluster et pour prendre en charge la propagation des balises de ressources.
- `ecs:StartTelemetrySession`— (Facultatif) Permet à l'agent de conteneur Amazon ECS de communiquer avec le plan de contrôle Amazon ECS afin de communiquer des informations et des mesures relatives à l'état de santé de chaque conteneur et de chaque tâche.

  Bien que cette autorisation ne soit pas obligatoire, nous vous recommandons de l'ajouter pour permettre aux métriques de l'instance de conteneur de démarrer des actions d'échelle et de recevoir des rapports relatifs aux commandes de vérification de l'état.
- `ecs:TagResource`— (Facultatif) Permet à l'agent de conteneur Amazon ECS de baliser le cluster lors de sa création et de baliser les instances de conteneur lorsqu'elles sont enregistrées dans un cluster.
- `ecs:UpdateContainerInstancesState` : permet au principal de modifier le statut d'une instance de conteneur Amazon ECS. Cette autorisation est utilisée par l'agent de conteneur Amazon ECS pour le drainage des instances Spot.
- `ecs:Submit*`— (Obligatoire) Cela inclut les actions `SubmitAttachmentStateChangesSubmitContainerStateChange`, et `SubmitTaskStateChange` API. Elles sont utilisées par l'agent de conteneur Amazon ECS pour signaler les modifications d'état pour chaque ressource au plan de contrôle Amazon ECS. L'`SubmitContainerStateChange`autorisation n'est plus utilisée par l'agent de conteneur Amazon ECS, mais elle est conservée pour garantir la rétrocompatibilité.

- `ecr:GetAuthorizationToken`— (Facultatif) Permet à un principal de récupérer un jeton d'autorisation. Un jeton d'autorisation représente vos informations d'authentification IAM et peut être utilisé pour accéder à n'importe quel registre Amazon ECR auquel votre principal IAM a accès. Le jeton d'autorisation reçu est valide pendant 12 heures.
- `ecr:BatchCheckLayerAvailability`— (Facultatif) Lorsqu'une image de conteneur est transférée vers un référentiel privé Amazon ECR, chaque couche d'image est vérifiée pour vérifier si elle est déjà envoyée. Le cas échéant, le calque d'image est ignoré.
- `ecr:GetDownloadUrlForLayer`— (Facultatif) Lorsqu'une image de conteneur est extraite d'un référentiel privé Amazon ECR, cette API est appelée une fois pour chaque couche d'image qui n'est pas déjà mise en cache.
- `ecr:BatchGetImage`— (Facultatif) Lorsqu'une image de conteneur est extraite d'un référentiel privé Amazon ECR, cette API est appelée une seule fois pour récupérer le manifeste de l'image.
- `logs:CreateLogStream`— (Facultatif) Permet à un principal de créer un flux de CloudWatch journaux pour un groupe de journaux spécifié.
- `logs:PutLogEvents`— (Facultatif) Permet à un principal de télécharger un lot d'événements de journal dans un flux de journal spécifié.

Rôles liés à un service

Vous pouvez utiliser le rôle lié à un service pour Amazon ECS afin d'autoriser le service Amazon ECS à appeler un autre service en votre APIs nom. Amazon ECS a besoin des autorisations pour créer et supprimer des interfaces réseau, enregistrer et annuler l'enregistrement des cibles auprès d'un groupe cible. Il doit également disposer des autorisations nécessaires pour créer et supprimer des stratégies de mise à l'échelle. Ces autorisations sont octroyées par le biais d'un rôle lié à un service. Ce rôle est créé en votre nom la première fois que vous utilisez le service.

> **Note**
>
> Si vous supprimez par inadvertance le rôle lié à un service, vous pouvez le recréer. Pour obtenir des instructions, veuillez consulter Création du rôle lié à un service.

Recommandations relatives aux rôles

Nous vous recommandons d'effectuer les opérations suivantes lorsque vous configurez les rôles et politiques IAM de votre tâche.

Bloquer l'accès aux EC2 métadonnées Amazon

Lorsque vous exécutez vos tâches sur des EC2 instances Amazon, nous vous recommandons vivement de bloquer l'accès aux EC2 métadonnées Amazon afin d'empêcher vos conteneurs d'hériter du rôle attribué à ces instances. Si vos applications doivent appeler une action d' AWS API, utilisez plutôt les rôles IAM pour les tâches.

Pour empêcher les tâches exécutées en mode pont d'accéder aux EC2 métadonnées Amazon, exécutez la commande suivante ou mettez à jour les données utilisateur de l'instance. Pour plus d'instructions sur la mise à jour des données utilisateur d'une instance, veuillez consulter cet article d'AWS Support. Pour plus d'informations sur le mode pont de définition des tâches, veuillez consulter Mode réseau de définition des tâches (langue française non garantie).

```
sudo yum install -y iptables-services; sudo iptables --insert FORWARD 1 --in-interface
 docker+ --destination 169.254.169.254/32 --jump DROP
```

Pour que cette modification soit maintenue après un redémarrage, exécutez la commande suivante, spécifique à votre Amazon Machine Image (AMI) :

- Amazon Linux 2

  ```
  sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now
   iptables
  ```

- Amazon Linux

  ```
  sudo service iptables save
  ```

Pour les tâches qui utilisent le mode réseau `awsvpc`, définissez la variable d'environnement `ECS_AWSVPC_BLOCK_IMDS` sur `true` dans le fichier `/etc/ecs/ecs.config`.

Vous devez définir la `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` variable sur `false` dans le `ecs-agent config` fichier pour empêcher les conteneurs exécutés sur le `host` réseau d'accéder aux EC2 métadonnées Amazon.

Utiliser le mode **`awsvpc`** réseau

Utilisez le mode réseau `awsvpc` pour limiter le flux de trafic entre les différentes tâches ou entre vos tâches et les autres services exécutés au sein de votre Amazon VPC. Cela ajoute une couche de sécurité supplémentaire. Le mode `awsvpc` réseau fournit une isolation réseau au niveau des tâches

pour les tâches exécutées sur Amazon. EC2 Il s'agit du mode activé par défaut AWS Fargate. C'est le seul mode réseau que vous pouvez utiliser pour attribuer un groupe de sécurité à des tâches.

Utiliser les dernières informations consultées pour affiner les rôles

Nous vous recommandons de supprimer toutes les actions qui n'ont jamais été utilisées ou qui n'ont pas été utilisées depuis un certain temps. Cela empêche tout accès indésirable. Pour ce faire, passez en revue les dernières informations consultées fournies par IAM, puis supprimez les actions qui n'ont jamais été utilisées ou qui ne l'ont pas été récemment. Les étapes de la section suivantes décrivent comment procéder.

Utilisez la commande suivante pour générer un rapport contenant les dernières informations d'accès pour la stratégie référencée :

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Utilisez le `JobId` de la sortie pour exécuter la commande suivante. Après cela, vous pouvez consulter les résultats du rapport.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Pour plus d'informations, voir Affiner les autorisations en AWS utilisant les dernières informations consultées.

Surveillez AWS CloudTrail les activités suspectes

Vous pouvez surveiller AWS CloudTrail toute activité suspecte. La plupart des appels d' AWS API sont enregistrés en AWS CloudTrail tant qu'événements. Ils sont analysés par AWS CloudTrail Insights et vous êtes alerté de tout comportement suspect associé aux appels d'`write`API. Cela peut inclure un pic du volume d'appels. Ces alertes incluent des informations telles que l'heure à laquelle l'activité inhabituelle s'est produite et le principal ARN d'identité qui a contribué à la APIs.

Vous pouvez identifier les actions effectuées par les tâches ayant un rôle IAM dans AWS CloudTrail examinant la propriété `userIdentity` de l'événement. Dans l'exemple suivant, l'`arn` comprend le nom du rôle endossé, `s3-write-go-bucket-role`, suivi du nom de la tâche, `7e9894e088ad416eb5cab92afExample`.

```
"userIdentity": {
    "type": "AssumedRole",
```

```
    "principalId": "AROA36C6WWEJ2YEXAMPLE:7e9894e088ad416eb5cab92afExample",
    "arn": "arn:aws:sts::123456789012:assumed-role/s3-write-go-bucket-
role/7e9894e088ad416eb5cab92afExample",
    ...
}
```

> **Note**
>
> Lorsque des tâches qui assument un rôle sont exécutées sur des instances de EC2 conteneur Amazon, une demande est enregistrée par l'agent de conteneur Amazon ECS dans le journal d'audit de l'agent situé à une adresse au `/var/log/ecs/audit.log.YYYY-MM-DD-HH` format. Pour plus d'informations, veuillez consulter Journal des rôles IAM pour les tâches et Journalisation des événements Insights pour les journaux de suivi.

Rôle IAM d'exécution de tâche Amazon ECS

Le rôle IAM d'exécution de tâche accorde aux agents de conteneur Amazon ECS et Fargate l'autorisation d'effectuer des appels d'API AWS en votre nom. Le rôle IAM d'exécution de la tâche est requis en fonction des besoins de votre tâche. Vous pouvez avoir plusieurs rôles d'exécution de tâche pour différents objectifs et services associés à votre compte.

> **Note**
>
> Ces autorisations sont mises à la disposition de l'agent exécuté sur votre instance par Amazon ECS qui lui envoie régulièrement les informations d'identification temporaires du rôle, mais elles ne sont pas directement accessibles par les conteneurs de la tâche. Pour connaître les autorisations IAM nécessaires à l'exécution du code de votre application dans le conteneur, consultezRôle IAM de la tâche Amazon ECS.

Voici des cas d'utilisation courants pour un rôle IAM d'exécution de tâche :

- Votre tâche est hébergée sur AWS Fargateou sur une instance externe et :
  - extrait une image de conteneur depuis un référentiel privé Amazon ECR.
  - extrait une image de conteneur d'un référentiel privé Amazon ECR dans un compte différent de celui qui exécute la tâche.

- envoie les journaux des conteneurs à CloudWatch Logs à l'aide du pilote de `awslogs` journal. Pour de plus amples informations, veuillez consulter Envoyez les journaux Amazon ECS à CloudWatch .
- Vos tâches sont hébergées sur des instances AWS Fargate ou sur EC2 des instances Amazon et :
  - utilise l'authentification du registre privé. Pour de plus amples informations, veuillez consulter Autorisations d'authentification du registre privé.
  - utilise la surveillance du temps d'exécution.
  - la définition de la tâche fait référence à des données sensibles à l'aide des secrets de Secrets Manager ou AWS des paramètres du magasin de paramètres de Systems Manager. Pour de plus amples informations, veuillez consulter Permissions de Secrets Manager ou de Systems Manager.

> **Note**
>
> Le rôle d'exécution de tâche est pris en charge par l'agent de conteneur Amazon ECS, version 1.16.0 et ultérieures.

Amazon ECS fournit la stratégie gérée nommée `AmazonECSTaskExecutionRolePolicy` qui contient les autorisations nécessaires pour les cas d'utilisation courants décrits ci-dessus. Pour plus d'informations, consultez Amazon ECSTask ExecutionRolePolicy dans le AWS Managed Policy Reference Guide. Il peut être nécessaire d'ajouter des politiques en ligne à votre rôle d'exécution de tâches pour des cas d'utilisation particuliers

La console Amazon ECS crée un rôle d'exécution de tâche. Vous pouvez joindre manuellement la politique IAM gérée aux tâches afin de permettre à Amazon ECS d'ajouter des autorisations pour les fonctionnalités et améliorations futures au fur et à mesure de leur introduction. Vous pouvez utiliser la recherche dans la console IAM pour rechercher `ecsTaskExecutionRole` et vérifier si votre compte possède déjà le rôle d'exécution des tâches. Pour plus d'informations, consultez la section Recherche dans la console IAM dans le guide de l'utilisateur d'IAM.

Si vous extrayez des images en tant qu'utilisateur authentifié, vous êtes moins susceptible d'être affecté par les modifications apportées à l'Docker Hubutilisation et aux limites. Pour plus d'informations, veuillez consulter Authentification de registre privé pour les instances de conteneur (langue française non garantie).

En utilisant Amazon ECR et Amazon ECR Public, vous pouvez éviter les limites imposées par Docker. Si vous extrayez des images depuis Amazon ECR, cela permet également de raccourcir les temps d'extraction du réseau et de réduire les modifications de transfert de données lorsque le trafic quitte votre VPC.

Lorsque vous utilisez Fargate, vous devez vous authentifier auprès d'un registre d'images privé à l'aide de `repositoryCredentials`. Il n'est pas possible de définir les variables d'environnement de l'agent de conteneur Amazon ECS `ECS_ENGINE_AUTH_TYPE` ou `ECS_ENGINE_AUTH_DATA`, ou de modifier le fichier `ecs.config` pour les tâches hébergées sur Fargate. Pour plus d'informations, veuillez consulter Authentification de registre privé pour les tâches (langue française non garantie).

Création du rôle d'exécution de tâche

Si votre compte ne possède pas encore de rôle d'exécution de tâches, suivez les étapes ci-dessous pour créer le rôle.

AWS Management Console

Pour créer le rôle de service pour Elastic Container Service (console IAM)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez Elastic Container Service, puis choisissez le cas d'utilisation d'Elastic Container Service Task.
5. Choisissez Suivant.
6. Dans la section Ajouter des autorisations, recherchez Amazon ECSTask ExecutionRolePolicy, puis sélectionnez la politique.
7. Choisissez Suivant.
8. Dans Nom du rôle, entrez ecsTaskExecutionRôle.
9. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

AWS CLI

Remplacez tout *`user input`* par vos propres informations.

1. Créez un fichier nommé `ecs-tasks-trust-policy.json` contenant la stratégie d'approbation à utiliser pour le rôle IAM. Le fichier doit contenir ce qui suit :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Créez un rôle IAM nommé `ecsTaskExecutionRole` à l'aide de la stratégie d'approbation créée à l'étape précédente.

```
aws iam create-role \
      --role-name ecsTaskExecutionRole \
      --assume-role-policy-document file://ecs-tasks-trust-policy.json
```

3. Associez la `AmazonECSTaskExecutionRolePolicy` politique AWS gérée au `ecsTaskExecutionRole` rôle.

```
aws iam attach-role-policy \
      --role-name ecsTaskExecutionRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy
```

Après avoir créé le rôle, ajoutez-y des autorisations supplémentaires pour les fonctionnalités suivantes.

Fonctionnalité	Autorisations supplémentaires
Extrayez des images de conteneurs depuis des registres privés extérieurs AWS (tels que Docker Hub, Quay.io ou votre propre registre	Autorisations d'authentification du registre privé

Fonctionnalité	Autorisations supplémentaires
privé) à l'aide des informations d'identification de Secrets Manager	
Transmettez des données sensibles avec Systems Manager ou Secrets Manager	Permissions de Secrets Manager ou de Systems Manager
Demandez aux tâches Fargate d'extraire les images Amazon ECR sur les points de terminaison de l'interface	Tâches Fargate extrayant des images Amazon ECR au-delà des autorisations des points de terminaison de l'interface
Fichiers de configuration de l'hôte dans un compartiment Amazon S3	Autorisations de stockage de fichiers Amazon S3
Configurer Container Insights pour afficher les événements du cycle de vie d'Amazon ECS	Autorisations requises pour activer les événements du cycle de vie d'Amazon ECS dans Container Insights
Afficher les événements relatifs au cycle de vie d'Amazon ECS dans Container Insights	Autorisations requises pour afficher les événements du cycle de vie Amazon ECS dans Container Insights

Autorisations d'authentification du registre privé

L'authentification par registre privé permet à vos tâches Amazon ECS d'extraire des images de conteneurs depuis des registres privés extérieurs AWS (tels que Docker Hub, Quay.io ou votre propre registre privé) qui nécessitent des informations d'authentification. Cette fonctionnalité utilise Secrets Manager pour stocker en toute sécurité vos informations d'identification de registre, qui sont ensuite référencées dans votre définition de tâche à l'aide du `repositoryCredentials` paramètre.

Pour plus d'informations sur la configuration de l'authentification du registre privé, consultez Utilisation d'images non AWS conteneurisées dans Amazon ECS.

Pour donner accès aux secrets qui contiennent vos informations d'identification de registre privées, ajoutez les autorisations suivantes en tant que politique intégrée au rôle d'exécution des tâches. Pour plus d'informations, consultez Ajout et suppression de politiques IAM.

- `secretsmanager:GetSecretValue`—Nécessaire pour récupérer les informations d'identification du registre privé auprès de Secrets Manager.
- `kms:Decrypt` : obligatoire uniquement si votre secret utilise une clé KMS personnalisée et non la clé par défaut. L'Amazon Resource Name (ARN) de votre clé personnalisée doit être ajouté en tant que ressource.

Voici un exemple de stratégie en ligne qui ajoute les autorisations.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:secretsmanager:<region>:<aws_account_id>:secret:secret_name",
                "arn:aws:kms:<region>:<aws_account_id>:key/key_id"
            ]
        }
    ]
}
```

Permissions de Secrets Manager ou de Systems Manager

L'autorisation d'autoriser l'agent du conteneur à extraire les ressources nécessaires AWS Systems Manager ou les ressources de Secrets Manager. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.

Utilisation de Secrets Manager

Pour fournir l'accès aux secrets Secrets Manager que vous créez, ajoutez manuellement l'autorisation suivante au rôle d'exécution de tâche. Pour plus d'informations sur la gestion des autorisations, consultez Ajout et suppression de stratégies IAM dans le Guide de l'utilisateur IAM.

- `secretsmanager:GetSecretValue` : obligatoire si vous faites référence à un secret Secrets Manager. Ajoute l'autorisation pour récupérer le secret depuis Secrets Manager.

L'exemple suivant de politique ajoute les autorisations requises.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
      ]
    }
  ]
}
```

Utilisation de Systems Manager

> **Important**
>
> Pour les tâches qui utilisent le type de EC2 lancement, vous devez utiliser la variable de configuration de l'agent ECS `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` pour utiliser cette fonctionnalité. Vous pouvez l'ajouter au fichier `./etc/ecs/ecs.config` lors de la création de l'instance de conteneur, ou vous pouvez l'ajouter à une instance existante, puis redémarrer l'agent ECS. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

Pour permettre l'accès aux paramètres du magasin de paramètres du gestionnaire de systèmes que vous créez, ajoutez manuellement les autorisations suivantes en tant que stratégie au rôle d'exécution de tâche. Pour plus d'informations sur la gestion des autorisations, consultez Ajout et suppression de stratégies IAM dans le Guide de l'utilisateur IAM.

- `ssm:GetParameters` : obligatoire lorsque vous référencez un paramètre Systems Manager Parameter Store dans une définition de tâche. Ajoute l'autorisation pour récupérer les paramètres de Systems Manager.
- `secretsmanager:GetSecretValue` : obligatoire si vous référencez directement un secret Secrets Manager ou si votre paramètre Systems Manager Parameter Store référence un secret

Secrets Manager dans une définition de tâche. Ajoute l'autorisation pour récupérer le secret depuis Secrets Manager.

- `kms:Decrypt` : obligatoire uniquement si votre secret utilise une clé managée client et non la clé par défaut. L'ARN de votre clé personnalisée doit être ajouté en tant que ressource. Ajoute l'autorisation pour déchiffrer la clé gérée par le client.

L'exemple suivant de politique ajoute les autorisations requises.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetParameters",
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:ssm:region:aws_account_id:parameter/parameter_name",
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name",
        "arn:aws:kms:region:aws_account_id:key/key_id"
      ]
    }
  ]
}
```

Tâches Fargate extrayant des images Amazon ECR au-delà des autorisations des points de terminaison de l'interface

Lors du lancement des tâches qui utilisent le type de lancement Fargate et qui extraient des images à partir d'Amazon ECR, lorsqu'Amazon ECR est configuré pour utiliser un point de terminaison d'un VPC d'interface, vous pouvez restreindre les droits d'accès des tâches à un VPC ou à un point de terminaison d'un VPC spécifique. Pour ce faire, créez un rôle d'exécution de tâche pour les tâches qui utilisent des clés de condition IAM.

Utilisez les clés de condition globales IAM suivantes pour restreindre l'accès à un VPC ou à un point de terminaison d'un VPC spécifique. Pour de plus amples informations, veuillez consulter Clés de contexte de condition globale AWS.

- `aws:SourceVpc` : restreint l'accès à un VPC spécifique. Vous pouvez limiter le VPC au VPC qui héberge la tâche et le point de terminaison.
- `aws:SourceVpce` : restreint l'accès à un point de terminaison d'un VPC spécifique.

La stratégie de rôle d'exécution de tâche suivante fournit un exemple d'ajout de clés de condition :

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "arn:aws:ec2:arn:aws::111122223333:vpc-
endpoint/vpce-0123456789abcdef0",
                    "arn:aws:ec2:arn:aws::111122223333:vpc/vpc-0123456789abcdef0"
                }
            }
        }
    ]
}
```

Autorisations Amazon ECR

Les autorisations suivantes sont requises lorsque vous devez extraire des images de conteneurs depuis des référentiels privés Amazon ECR. Le rôle d'exécution des tâches doit disposer de ces autorisations pour permettre au conteneur Amazon ECS et aux agents Fargate d'extraire les images du conteneur en votre nom. Pour les implémentations ECS de base, ces autorisations doivent être ajoutées au rôle d'exécution des tâches plutôt qu'au rôle IAM des tâches.

La politique gérée du rôle d'exécution des tâches Amazon ECS (`AmazonECSTaskExecutionRolePolicy`) inclut les autorisations nécessaires pour extraire des images depuis Amazon ECR. Si vous utilisez la politique gérée, il n'est pas nécessaire d'ajouter ces autorisations séparément.

Si vous créez une politique personnalisée, incluez les autorisations suivantes pour autoriser l'extraction d'images depuis Amazon ECR :

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Notez que ces autorisations sont différentes de celles qui peuvent être requises dans le rôle IAM de tâche si le code de votre application doit interagir directement avec Amazon ECR. APIs Pour plus d'informations sur les autorisations de rôle IAM des tâches pour Amazon ECR, consultez. Autorisations Amazon ECR

Autorisations de stockage de fichiers Amazon S3

Lorsque vous spécifiez un fichier de configuration hébergé dans Amazon S3, le rôle d'exécution de la tâche doit inclure l'`s3:GetObject`autorisation pour le fichier de configuration et l'`s3:GetBucketLocation`autorisation sur le compartiment Amazon S3 dans lequel se trouve le fichier. Pour plus d'informations, consultez la section Actions politiques pour Amazon S3 dans le guide de l'utilisateur d'Amazon Simple Storage Service.

L'exemple de politique suivant ajoute les autorisations requises pour récupérer un fichier depuis Amazon S3. Spécifiez le nom de votre compartiment Amazon S3 et le nom du fichier de configuration.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/folder_name/config_file_name"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket"
      ]
    }
  ]
}
```

Rôle IAM de la tâche Amazon ECS

Vos tâches Amazon ECS peuvent être associées à un rôle IAM. Les autorisations accordées dans le rôle IAM sont accordées aux conteneurs exécutés dans le cadre de la tâche. Ce rôle permet au code de votre application (exécuté dans le conteneur) d'utiliser d'autres AWS services. Le rôle de tâche est requis lorsque votre application accède à d'autres AWS services, tels qu'Amazon S3.

> **Note**
>
> Le conteneur Amazon ECS et les agents Fargate n'accèdent pas à ces autorisations. Pour connaître les autorisations IAM dont Amazon ECS a besoin pour extraire des images de conteneurs et exécuter la tâche, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

Les avantages de l'utilisation des rôles de tâches sont les suivants :

- Séparation des problèmes : si vous utilisez le type de EC2 lancement, les rôles IAM des tâches vous permettent de spécifier des autorisations IAM pour vos conteneurs sans que ces autorisations soient spécifiées à l'aide de profils d' EC2 instance (pour plus d'informations, voir Utilisation des profils d'instance dans le Guide de l'AWS Identity and Access Management utilisateur). Par conséquent, vous pouvez déployer vos applications de manière indépendante et uniforme sur les instances de conteneur ECS sans avoir à modifier les autorisations IAM associées EC2 aux instances.
- Auditabilité : l'enregistrement des accès et des événements est disponible CloudTrail pour garantir un audit rétrospectif. Les informations d'identification des tâches ont un contexte `taskArn` « » associé à la session, de sorte que les CloudTrail journaux indiquent à quelle tâche les informations d'identification du rôle ont été attribuées.
- Fourniture d'informations d'identification uniforme : ECS fournit des informations d'identification de rôle IAM à vos conteneurs et les rend accessibles via une interface bien définie, quel que soit le type de lancement associé à vos tâches. Sur ECS Fargate EC2 , les profils d'instance ne sont pas disponibles pour les conteneurs de vos tâches. Les rôles Task IAM vous permettent d'associer des autorisations IAM à vos conteneurs quel que soit le type de lancement lorsque vous utilisez un AWS SDK ou une AWS CLI dans vos conteneurs. Pour plus d'informations sur la manière dont le AWS SDK accède à ces informations d'identification, consultez la section Fournisseur d'informations d'identification du conteneur.

> **Important**
>
> Les conteneurs ne constituent pas une limite de sécurité et l'utilisation de rôles IAM de tâches ne change rien à cela. Chaque tâche exécutée sur Fargate possède sa propre limite d'isolation et ne partage pas le noyau sous-jacent, les ressources du processeur, les ressources de mémoire ou l'interface elastic network avec une autre tâche. Pour les instances de conteneur externes EC2 et externes sur ECS, il n'y a pas d'isolation des tâches (contrairement à Fargate) et les conteneurs peuvent potentiellement accéder aux informations d'identification pour d'autres tâches sur la même instance de conteneur. Ils peuvent également accéder aux autorisations attribuées au rôle d'instance de conteneur ECS. Suivez les recommandations de la section Recommandations relatives aux rôles pour bloquer l'accès au service de métadonnées d' EC2 instance Amazon pour les conteneurs (pour plus d'informations, consultez Utiliser le service de métadonnées d'instance pour accéder aux métadonnées d'instance dans le guide de EC2 l'utilisateur Amazon). Notez que lorsque vous spécifiez un rôle IAM pour une tâche, la AWS CLI ou un autre rôle SDKs dans les conteneurs de cette tâche utilisent exclusivement les AWS informations d'identification fournies par le rôle de tâche et n'héritent d'aucune autorisation IAM de l'instance Amazon EC2 ou externe sur laquelle ils s'exécutent.

Création du rôle IAM de la tâche

Lorsque vous créez une politique IAM pour vos tâches, celle-ci doit inclure les autorisations que vous souhaitez que les conteneurs de vos tâches assument. Vous pouvez utiliser une politique AWS gérée existante ou créer une politique personnalisée à partir de zéro qui répond à vos besoins spécifiques. Pour plus d’informations, consultez Création de politiques IAM dans le Guide de l’utilisateur IAM.

> **Important**
>
> Pour les tâches Amazon ECS (pour tous les types de lancement), nous vous recommandons d'utiliser la politique et le rôle IAM pour vos tâches. Ces informations d'identification permettent à votre tâche d'effectuer des demandes d' AWS API sans appeler `sts:AssumeRole` pour assumer le même rôle que celui déjà associé à la tâche. Si votre tâche nécessite qu'un rôle s'assume, vous devez créer une politique de confiance qui autorise explicitement ce rôle à s'assumer. Pour plus d'informations, consultez la section Mise à jour d'une politique d'approbation des rôles dans le Guide de l'utilisateur IAM.

Après la création de la politique IAM, vous pouvez créer un rôle IAM qui inclut la politique que vous référencez dans votre définition de tâche Amazon ECS. Vous pouvez créer le rôle à l'aide du cas d'utilisation Amazon Elastic Container Service Task dans la console IAM. Vous pouvez ensuite associer votre politique IAM spécifique au rôle qui donne aux conteneurs de votre tâche les autorisations souhaitées. Les procédures ci-dessous expliquent comment procéder.

Si vous avez plusieurs définitions de tâche ou services qui requièrent des autorisations IAM, nous vous conseillons de créer un rôle pour chaque définition de tâche ou service en lui affectant les autorisations minimales requises pour les tâches à exécuter, afin de minimiser les accès octroyés à chaque tâche.

Pour plus d'informations sur le point de terminaison de service de votre région, consultez la section Points de terminaison du service dans le Référence générale d'Amazon Web Services Guide.

Le rôle de tâche IAM doit posséder une politique de confiance qui spécifie le service `ecs-tasks.amazonaws.com`. L'`sts:AssumeRole`autorisation permet à vos tâches d'assumer un rôle IAM différent de celui utilisé EC2 par l'instance Amazon. Ainsi, votre tâche n'hérite pas du rôle associé à l' EC2 instance Amazon. Voici un exemple de politique de confiance. Remplacez l'identifiant de région et spécifiez le numéro de AWS compte que vous utilisez lors du lancement des tâches.

> **Important**
>
> Lorsque vous créez votre rôle IAM de tâche, il est recommandé d'utiliser les clés de `aws:SourceArn` condition `aws:SourceAccount` ou de la politique de relation de confiance associée au rôle afin de définir davantage les autorisations afin d'éviter tout problème de sécurité adjoint confus. L'utilisation de la clé de condition `aws:SourceArn` pour spécifier un cluster spécifique n'est actuellement pas prise en charge, vous devez utiliser le caractère générique pour spécifier tous les clusters. Pour en savoir plus sur le problème des députés confus et sur la manière de protéger votre AWS compte, consultez la section Le problème des adjoints confus dans le guide de l'utilisateur d'IAM.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
```

```
            "Effect":"Allow",
            "Principal":{
               "Service":[
                  "ecs-tasks.amazonaws.com"
               ]
            },
            "Action":"sts:AssumeRole",
            "Condition":{
               "ArnLike":{
               "aws:SourceArn":"arn:aws:ecs:us-west-2:111122223333:*"
               },
               "StringEquals":{
                  "aws:SourceAccount":"111122223333"
               }
            }
         }
      ]
}
```

Utilisez la procédure suivante pour créer une politique permettant de récupérer des objets depuis Amazon S3 avec un exemple de politique. Remplacez tous les *user input* par vos propres valeurs.

AWS Management Console

Pour utiliser l'éditeur de politique JSON afin de créer une politique

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).

   Si vous sélectionnez Politiques pour la première fois, la page Bienvenue dans les politiques gérées s'affiche. Sélectionnez Mise en route.
3. En haut de la page, sélectionnez Créer une politique.
4. Dans la section Éditeur de politique, choisissez l'option JSON.
5. Entrez le document de politique JSON suivant :

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
```

```
            "Effect":"Allow",
            "Action":[
               "s3:GetObject"
            ],
            "Resource":[
               "arn:aws:s3:::my-task-secrets-bucket/*"
            ]
         }
      ]
   }
```

6. Choisissez Suivant.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez la page Restructuration de politique dans le Guide de l'utilisateur IAM.

7. Sur la page Vérifier et créer, saisissez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique.
8. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

AWS CLI

Remplacez tous les *`user input`* par vos propres valeurs.

1. Créez un fichier nommé `s3-policy.json` avec le contenu suivant.

   JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
```

```
            "s3:GetObject"
         ],
         "Resource":[
            "arn:aws:s3:::my-task-secrets-bucket/*"
         ]
      }
   ]
}
```

2. Utilisez la commande suivante pour créer la stratégie IAM à l'aide du fichier de document de stratégie JSON. Remplacez tous les `user input` par vos propres valeurs.

```
aws iam create-policy \
      --policy-name taskRolePolicy \
      --policy-document file://s3-policy.json
```

Utilisez la procédure suivante pour créer le rôle de service.

AWS Management Console

Pour créer le rôle de service pour Elastic Container Service (console IAM)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez Elastic Container Service, puis choisissez le cas d'utilisation d'Elastic Container Service Task.
5. Choisissez Suivant.
6. Pour Ajouter des autorisations, recherchez et choisissez la politique que vous avez créée.
7. Choisissez Suivant.
8. Dans le champ Role name (Nom de rôle), saisissez un nom pour votre rôle. Pour cet exemple, tapez `AmazonECSTaskS3BucketRole` comme nom du rôle.
9. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

AWS CLI

1. Créez un fichier nommé `ecs-tasks-trust-policy.json` contenant la politique de confiance à utiliser pour le rôle IAM de la tâche. Le fichier doit contenir les éléments suivants. Remplacez l'identifiant de région et spécifiez le numéro de AWS compte que vous utilisez lors du lancement des tâches.

   JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":[
               "ecs-tasks.amazonaws.com"
            ]
         },
         "Action":"sts:AssumeRole",
         "Condition":{
            "ArnLike":{
            "aws:SourceArn":"arn:aws:ecs:us-west-2:111122223333:*"
            },
            "StringEquals":{
               "aws:SourceAccount":"111122223333"
            }
         }
      }
   ]
}
```

2. Créez un rôle IAM nommé `ecsTaskRole` à l'aide de la stratégie d'approbation créée à l'étape précédente.

```
aws iam create-role \
      --role-name ecsTaskRole \
      --assume-role-policy-document file://ecs-tasks-trust-policy.json
```

3. Récupérez l'ARN de la politique IAM que vous avez créée à l'aide de la commande suivante. *`taskRolePolicy`*Remplacez-le par le nom de la politique que vous avez créée.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`taskRolePolicy`].Arn'
```

4. Attachez la politique IAM que vous avez créée au `ecsTaskRole` rôle. Remplacez le `policy-arn` par l'ARN de la politique que vous avez créée.

```
aws iam attach-role-policy \
      --role-name ecsTaskRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/taskRolePolicy
```

Après avoir créé le rôle, ajoutez-y des autorisations supplémentaires pour les fonctionnalités suivantes.

Fonctionnalité	Autorisations supplémentaires
Utiliser ECS Exec	Autorisations ECS Exec
Utiliser une image provenant d'un référentiel Amazon ECR privé	Autorisations Amazon ECR
Utiliser EC2 des instances (Windows et Linux)	Configuration supplémentaire des EC2 instances Amazon
Utiliser des instances externes	Configuration supplémentaire de l'instance externe
Utiliser des EC2 instances Windows	Configuration supplémentaire de l'instance Amazon EC2 Windows

Autorisations Amazon ECR

Les autorisations suivantes sont requises lorsque le code de votre application doit interagir directement avec les référentiels Amazon ECR. Notez que pour une implémentation de base où vous devez uniquement extraire des images depuis Amazon ECR, ces autorisations ne sont pas requises au niveau du rôle IAM de la tâche. Au lieu de cela, le rôle d'exécution de tâches Amazon ECS doit disposer de ces autorisations. Pour plus d'informations sur le rôle d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Si le code de votre application exécuté dans le conteneur doit interagir APIs directement avec Amazon ECR, vous devez ajouter les autorisations suivantes à un rôle IAM de tâche et inclure le rôle IAM de tâche dans votre définition de tâche. Pour plus d'informations, consultez la section Ajout et suppression de politiques IAM dans le guide de l'utilisateur IAM.

Utilisez la politique suivante pour votre rôle IAM de tâche afin d'ajouter les autorisations Amazon ECR requises pour les applications de conteneur qui doivent interagir directement avec Amazon ECR :

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Autorisations ECS Exec

La fonctionnalité ECS Exec nécessite un rôle IAM de tâche pour accorder aux conteneurs les autorisations nécessaires à la communication entre l'agent SSM géré (`execute-command`agent) et le service SSM. Vous devez ajouter les autorisations suivantes à un rôle IAM de tâche et inclure le rôle IAM de tâche dans votre définition de tâche. Pour plus d'informations, consultez la section Ajout et suppression de politiques IAM dans le guide de l'utilisateur IAM.

Utilisez la stratégie suivante pour votre rôle IAM de tâche afin d'ajouter les autorisations SSM requises.

JSON

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
        "Effect": "Allow",
        "Action": [
             "ssmmessages:CreateControlChannel",
             "ssmmessages:CreateDataChannel",
             "ssmmessages:OpenControlChannel",
             "ssmmessages:OpenDataChannel"
        ],
       "Resource": "*"
       }
    ]
}
```

Configuration supplémentaire des EC2 instances Amazon

Nous vous recommandons de limiter les autorisations dans votre rôle d'instance de conteneur à la liste minimale des autorisations utilisées dans la politique IAM gérée `AmazonEC2ContainerServiceforEC2Role`.

Vos EC2 instances Amazon nécessitent au moins une version `1.11.0` de l'agent de conteneur pour utiliser le rôle de tâche ; nous vous recommandons toutefois d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS. Si vous utilisez une AMI optimisée pour Amazon ECS, votre instance a besoin d'au moins une partie `1.11.0-1` du `ecs-init` package. Si vos instances utilisent la dernière AMI optimisée pour Amazon ECS, elles contiennent les versions requises de l'agent de conteneur et `ecs-init`. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.

Si vous n'utilisez pas l'AMI optimisée pour Amazon ECS pour vos instances de conteneur, ajoutez l'`--net=host`option à votre docker run commande qui démarre l'agent et les variables de configuration d'agent suivantes pour la configuration souhaitée (pour plus d'informations, consultezConfiguration de l'agent de conteneur Amazon ECS) :

`ECS_ENABLE_TASK_IAM_ROLE=true`

Utilise les rôles IAM pour les tâches des conteneurs associés aux modes réseau `bridge` et `default`.

`ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true`

Utilise les rôles IAM pour les tâches des conteneurs associés au mode réseau `host`. Cette variable est uniquement prise en charge sur les versions d'agent 1.12.0 et ultérieures.

Pour obtenir un exemple de commande d'exécution, consultez Mise à jour manuelle de l'agent de conteneur Amazon ECS (pour les applications non optimisées pour Amazon ECS AMIs). Vous devrez également définir les commandes réseau suivantes sur votre instance de conteneur afin que les conteneurs de vos tâches puissent récupérer leurs AWS informations d'identification :

```
sudo sysctl -w net.ipv4.conf.all.route_localnet=1
sudo iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
sudo iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Pour garantir le maintien de ces règles iptables après un redémarrage, vous devez les enregistrer sur votre instance de conteneur. Vous pouvez utiliser les commandes iptables-save et iptables-restore pour enregistrer vos règles iptables et les restaurer au démarrage. Pour plus d'informations, consultez la documentation de votre système d'exploitation.

Pour empêcher les conteneurs exécutés par des tâches utilisant le mode `awsvpc` réseau d'accéder aux informations d'identification fournies au profil d' EC2 instance Amazon, tout en autorisant les autorisations fournies par le rôle de tâche, définissez la variable de configuration de l'`ECS_AWSVPC_BLOCK_IMDS`agent sur `true` dans le fichier de configuration de l'agent et redémarrez l'agent. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

Pour empêcher les conteneurs exécutés par des tâches utilisant le mode `bridge` réseau d'accéder aux informations d'identification fournies au profil d' EC2 instance Amazon, tout en autorisant les autorisations fournies par le rôle de tâche, exécutez la iptables commande suivante sur vos EC2 instances Amazon. Cette commande n'a pas d'incidence sur les conteneurs dans les tâches utilisant le mode réseau `host` ou `awsvpc`. Pour de plus amples informations, veuillez consulter Mode réseau.

- ```
  sudo yum install -y iptables-services; sudo iptables --insert DOCKER-USER 1 --in-
  interface docker+ --destination 169.254.169.254/32 --jump DROP
  ```

  Vous devez enregistrer cette iptables règle sur votre EC2 instance Amazon pour qu'elle puisse survivre au redémarrage. Lors de l'utilisation de l'AMI optimisée pour Amazon ECS, vous pouvez
  ```

utiliser la commande qui suit. Pour les autres systèmes d'exploitation, consultez la documentation correspondante.

```
sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now
 iptables
```

Configuration supplémentaire de l'instance externe

Vos instances externes ont besoin d'au moins une version `1.11.0` de l'agent de conteneur pour utiliser les rôles IAM des tâches ; nous vous recommandons toutefois d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS. Si vous utilisez l'AMI Linux optimisée pour Amazon ECS, votre instance doit également au moins disposer de la version `1.11.0-1` du package `ecs-init`. Si vos instances utilisent la dernière AMI optimisée pour Amazon ECS, elles contiennent les versions requises de l'agent de conteneur et `ecs-init`. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.

Si vous n'utilisez pas l'AMI optimisée pour Amazon ECS pour vos instances de conteneur, ajoutez l'`--net=host`option à votre docker run commande qui démarre l'agent et les variables de configuration d'agent suivantes pour la configuration souhaitée (pour plus d'informations, consultezConfiguration de l'agent de conteneur Amazon ECS) :

`ECS_ENABLE_TASK_IAM_ROLE=true`

Utilise les rôles IAM pour les tâches des conteneurs associés aux modes réseau `bridge` et `default`.

`ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true`

Utilise les rôles IAM pour les tâches des conteneurs associés au mode réseau `host`. Cette variable est uniquement prise en charge sur les versions d'agent 1.12.0 et ultérieures.

Pour obtenir un exemple de commande d'exécution, consultez Mise à jour manuelle de l'agent de conteneur Amazon ECS (pour les applications non optimisées pour Amazon ECS AMIs). Vous devrez également définir les commandes réseau suivantes sur votre instance de conteneur afin que les conteneurs de vos tâches puissent récupérer leurs AWS informations d'identification :

```
sudo sysctl -w net.ipv4.conf.all.route_localnet=1
```

```
sudo iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
sudo iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Pour garantir le maintien de ces règles iptables après un redémarrage, vous devez les enregistrer sur votre instance de conteneur. Vous pouvez utiliser les commandes iptables-save et iptables-restore pour enregistrer vos règles iptables et les restaurer au démarrage. Pour plus d'informations, consultez la documentation de votre système d'exploitation.

Configuration supplémentaire de l'instance Amazon EC2 Windows

> **Important**
>
> Cela s'applique uniquement aux conteneurs Windows EC2 qui utilisent des rôles de tâche.

Le rôle de tâche associé aux fonctionnalités Windows nécessite une configuration supplémentaire activée EC2.

- Lorsque vous lancez vos instances de conteneur, vous devez définir l'option `-EnableTaskIAMRole` dans le script de données utilisateur des instances de conteneur. `EnableTaskIAMRole` active la fonction Rôles IAM pour les tâches. Par exemple :

  ```
  <powershell>
  Import-Module ECSTools
  Initialize-ECSAgent -Cluster 'windows' -EnableTaskIAMRole
  </powershell>
  ```

- Vous devez amorcer votre conteneur avec les commandes de mise en réseau fournies dans Script d'amorçage du conteneur Amazon ECS.
- Vous devez créer un rôle et une stratégie IAM pour vos tâches. Pour de plus amples informations, veuillez consulter Création du rôle IAM de la tâche.
- Les rôles IAM pour le fournisseur d'informations d'identification de tâche utilisent le port 80 sur l'instance de conteneur. Par conséquent, si vous configurez des rôles IAM pour les tâches sur votre instance de conteneur, vos conteneurs ne peuvent pas utiliser le port 80 pour le port hôte des mappages de ports. Pour exposer vos conteneurs sur le port 80, nous vous recommandons de configurer un service pour ceux qui utilisent l'équilibrage de charge. Vous pouvez utiliser le port 80 sur l'équilibreur de charge. Ce faisant, le trafic peut être acheminé vers un autre port hôte sur vos

instances de conteneur. Pour de plus amples informations, veuillez consulter Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS.

- Si votre instance Windows est redémarrée, vous devez supprimer l'interface proxy et initialiser à nouveau l'agent conteneur Amazon ECS pour rétablir la sauvegarde du proxy d'informations d'identification.

Script d'amorçage du conteneur Amazon ECS

Avant que les conteneurs puissent accéder au proxy d'informations d'identification sur l'instance de conteneur afin d'obtenir des informations d'identification, le conteneur doit être amorcé avec les commandes de mise en réseau requises. L'exemple de script de code suivant doit être exécuté sur vos conteneurs lorsqu'ils démarrent.

> **Note**
>
> Vous n'avez pas besoin d'exécuter ce script lorsque vous utilisez le mode réseau `awsvpc` sur Windows.

Si vous exécutez des conteneurs Windows qui incluent Powershell, utilisez le script suivant :

```
# Copyright Amazon.com Inc. or its affiliates. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License"). You may
# not use this file except in compliance with the License. A copy of the
# License is located at
#
# http://aws.amazon.com/apache2.0/
#
# or in the "license" file accompanying this file. This file is distributed
# on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
# express or implied. See the License for the specific language governing
# permissions and limitations under the License.

$gateway = (Get-NetRoute | Where { $_.DestinationPrefix -eq '0.0.0.0/0' } | Sort-Object
 RouteMetric | Select NextHop).NextHop
$ifIndex = (Get-NetAdapter -InterfaceDescription "Hyper-V Virtual Ethernet*" | Sort-
Object | Select ifIndex).ifIndex
New-NetRoute -DestinationPrefix 169.254.170.2/32 -InterfaceIndex $ifIndex -NextHop
 $gateway -PolicyStore ActiveStore # credentials API
```

```
New-NetRoute -DestinationPrefix 169.254.169.254/32 -InterfaceIndex $ifIndex -NextHop
 $gateway -PolicyStore ActiveStore # metadata API
```

Si vous exécutez des conteneurs Windows qui n'ont que le shell Command, utilisez le script suivant :

```
# Copyright Amazon.com Inc. or its affiliates. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License"). You may
# not use this file except in compliance with the License. A copy of the
# License is located at
#
# http://aws.amazon.com/apache2.0/
#
# or in the "license" file accompanying this file. This file is distributed
# on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
# express or implied. See the License for the specific language governing
# permissions and limitations under the License.

for /f "tokens=1" %i  in ('netsh interface ipv4 show interfaces ^| findstr /x /r
 ".*vEthernet.*"') do set interface=%i
for /f "tokens=3" %i  in ('netsh interface ipv4 show addresses %interface% ^| findstr /
x /r ".*Default.Gateway.*"') do set gateway=%i
netsh interface ipv4 add route prefix=169.254.170.2/32 interface="%interface%"
 nexthop="%gateway%" store=active # credentials API
netsh interface ipv4 add route prefix=169.254.169.254/32 interface="%interface%"
 nexthop="%gateway%" store=active # metadata API
```

Rôle IAM d'instance de conteneur Amazon ECS

Les instances de conteneur Amazon ECS, y compris les instances Amazon EC2 et externes, exécutent l'agent de conteneur Amazon ECS et nécessitent un rôle IAM pour que le service sache que l'agent vous appartient. Avant de pouvoir lancer des instances de conteneur et de les enregistrer dans un cluster, vous devez créer le rôle IAM qu'elles utiliseront. Le rôle est créé dans le compte que vous utilisez pour vous connecter à la console ou exécuter les AWS CLI commandes.

> **Important**
>
> Si vous enregistrez des instances externes dans votre cluster, le rôle IAM que vous utilisez nécessite également des autorisations Systems Manager. Pour de plus amples informations, veuillez consulter Rôle IAM dans Amazon ECS Anywhere.

Amazon ECS fournit le fichier de stratégie IAM gérée `AmazonEC2ContainerServiceforEC2Role` qui contient les autorisations nécessaires à l'utilisation de l'ensemble complet de fonctions Amazon ECS. Cette stratégie gérée peut être attachée à un rôle IAM et associée à vos instances de conteneur. Sinon, vous pouvez utiliser la stratégie gérée comme guide lorsque vous créez une stratégie personnalisée à utiliser. Le rôle d'instance de conteneur fournit les autorisations nécessaires pour que l'agent de conteneur Amazon ECS et le daemon Docker puissent appeler en votre AWS APIs nom. Pour de plus amples informations sur la stratégie gérée, veuillez consulter EC2ContainerServiceforEC2Rôle Amazon.

Création du rôle d'instance de conteneur

> **Important**
>
> Si vous enregistrez des instances externes dans votre cluster, veuillez consulter Rôle IAM dans Amazon ECS Anywhere.

Vous pouvez créer et associer manuellement le rôle à la politique IAM gérée pour les instances de conteneur afin de permettre à Amazon ECS d'ajouter des autorisations pour les futures fonctionnalités et améliorations au fur et à mesure qu'elles sont introduites. Utilisez la procédure suivante pour joindre la politique IAM gérée si nécessaire.

AWS Management Console

Pour créer le rôle de service pour Elastic Container Service (console IAM)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez Elastic Container Service, puis choisissez le EC2 rôle pour le cas d'utilisation d'Elastic Container Service.
5. Choisissez Suivant.
6. Dans la section Politiques d'autorisations, vérifiez que la politique Amazon EC2 ContainerServicefor EC2 Role est sélectionnée.

> **Important**
>
> La politique gérée par Amazon EC2 ContainerServicefor EC2 Role doit être attachée au rôle IAM de l'instance de conteneur, sinon vous recevrez une erreur en utilisant le AWS Management Console pour créer des clusters.

7. Choisissez Suivant.
8. Pour Nom du rôle, entrez ecsInstanceRole
9. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

AWS CLI

Remplacez tous les *`user input`* par vos propres valeurs.

1. Créez un fichier nommé `instance-role-trust-policy.json` avec le contenu suivant.

   JSON

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": { "Service": "ec2.amazonaws.com"},
         "Action": "sts:AssumeRole"
       }
     ]
   }
   ```

2. Utilisez la commande suivante pour créer le rôle IAM d'instance à l'aide du document de politique de confiance.

   ```
   aws iam create-role \
       --role-name ecsInstanceRole \
       --assume-role-policy-document file://instance-role-trust-policy.json
   ```

3. Créez un profil d'instance nommé `ecsInstanceRole-profile` à l'aide de la create-instance-profilecommande.

```
aws iam create-instance-profile --instance-profile-name ecsInstanceRole-profile
```

Exemple de réponse

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJTLBPJLEGREXAMPLE",
        "Roles": [],
        "CreateDate": "2022-04-12T23:53:34.093Z",
        "InstanceProfileName": "ecsInstanceRole-profile",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/ecsInstanceRole-
profile"
    }
}
```

4. Ajoutez le rôle *ecsInstanceRole* au profil d'instance *ecsInstanceRole-profile*.

```
aws iam add-role-to-instance-profile \
    --instance-profile-name ecsInstanceRole-profile \
    --role-name ecsInstanceRole
```

5. Associez la politique `AmazonEC2ContainerServiceRoleForEC2Role` gérée au rôle à l'aide de la commande suivante.

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role \
    --role-name ecsInstanceRole
```

Après avoir créé le rôle, ajoutez-y des autorisations supplémentaires pour les fonctionnalités suivantes.

Fonctionnalité	Autorisations supplémentaires
Amazon ECR contient l'image du conteneur	Autorisations Amazon ECR

Fonctionnalité	Autorisations supplémentaires
Demandez à CloudWatch Logs de surveiller les instances de conteneur	Surveillance des autorisations des instances de conteneurs
Fichiers de configuration de l'hôte dans un compartiment Amazon S3	Accès en lecture seule à Amazon S3

Autorisations Amazon ECR

Le rôle d'instance de conteneur Amazon ECS que vous utilisez avec vos instances de conteneur doit disposer des autorisations de politique IAM suivantes pour Amazon ECR.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Si vous utilisez la stratégie gérée `AmazonEC2ContainerServiceforEC2Role` pour vos instances de conteneurs, votre rôle possède les autorisations appropriées. Pour vérifier si votre rôle prend en charge Amazon ECR, consultez Rôle IAM d'instance de conteneur Amazon ECS dans le Guide du développeur de service de conteneur Amazon Elastic.

Autorisations requises pour définir le paramètre du compte AWSvpcTrunking

Amazon ECS prend en charge le lancement d'instances de conteneur avec une densité ENI accrue à l'aide des types d' EC2 instances Amazon pris en charge. Lorsque vous utilisez cette fonctionnalité, nous vous recommandons de créer deux rôles d'instance de conteneur. Activez le paramètre du `awsvpcTrunking` compte pour un rôle et utilisez ce rôle pour les tâches nécessitant une jonction ENI. Pour plus d'informations sur le paramétrage du `awsvpcTrunking` compte, consultez Accédez aux fonctionnalités d'Amazon ECS avec les paramètres du compte.

Le rôle d'instance de conteneur que vous utilisez avec vos instances de conteneur doit disposer des autorisations de politique IAM suivantes pour définir le paramètre du compte

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ListAccountSettings",
                "ecs:ListAttributes",
                "ecs:PutAccountSetting"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour utiliser le rôle d'instance de conteneur, ajoutez ce qui suit aux données utilisateur de l'instance :

```
#!/bin/bash
aws ecs put-account-setting --name awsvpcTrunking --value enabled --region region
ECS_CLUSTER=MyCluster>> /etc/ecs/ecs.config
EOF
```

Pour plus d'informations sur l'ajout de données utilisateur à vos EC2 instances, consultez la section Exécuter des commandes sur votre instance Linux au lancement dans le guide de EC2 l'utilisateur Amazon.

Accès en lecture seule à Amazon S3

Stocker les informations de configuration dans un compartiment privé d'Amazon S3 et accorder un accès en lecture seule au rôle IAM de votre instance de conteneur est un moyen pratique et sécurisé d'autoriser la configuration d'une instance de conteneur au moment du lancement. Vous pouvez stocker une copie de votre `ecs.config` fichier dans un compartiment privé, utiliser les données EC2 utilisateur Amazon pour l'installer, AWS CLI puis copier vos informations de configuration au `/etc/ecs/ecs.config` moment du lancement de l'instance.

Pour plus d'informations sur la création d'un fichier `ecs.config`, son stockage dans Amazon S3 et le lancement d'instances avec cette configuration, veuillez consulter Stockage de la configuration de l'instance de conteneur Amazon ECS dans Amazon S3.

Vous pouvez utiliser la AWS CLI commande suivante pour autoriser l'accès en lecture seule à Amazon S3 pour votre rôle d'instance de conteneur. *`ecsInstanceRole`*Remplacez-le par le nom du rôle que vous avez créé.

```
aws iam attach-role-policy \
      --role-name ecsInstanceRole \
      --policy-arn arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess
```

Vous pouvez également utiliser la console IAM pour ajouter un accès en lecture seule (`AmazonS3ReadOnlyAccess`) à Amazon S3 à votre rôle. Pour plus d'informations, consultez la section Mise à jour des autorisations pour un rôle dans le Guide de AWS Identity and Access Management l'utilisateur.

Autorisations requises pour définir le paramètre du compte AWSvpcTrunking

Le rôle d'instance de conteneur Amazon ECS que vous utilisez avec vos instances de conteneur doit disposer des autorisations de politique IAM suivantes pour définir le paramètre du compte

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ListAccountSettings",
```

```
                "ecs:ListAttributes",
                "ecs:PutAccountSetting"
            ],
            "Resource": "*"
        }
    ]
}
```

Surveillance des autorisations des instances de conteneurs

Avant que vos instances de conteneur puissent envoyer des données de journal à CloudWatch Logs, vous devez créer une politique IAM permettant à l'agent Amazon ECS d'y écrire les journaux d'applications du client CloudWatch (normalement gérés par le `awslogs` pilote). Après avoir créé la stratégie, attachez-la à`ecsInstanceRole`.

AWS Management Console

Pour utiliser l'éditeur de politique JSON afin de créer une politique

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).

   Si vous sélectionnez Politiques pour la première fois, la page Bienvenue dans les politiques gérées s'affiche. Sélectionnez Mise en route.
3. En haut de la page, sélectionnez Créer une politique.
4. Dans la section Éditeur de politique, choisissez l'option JSON.
5. Entrez le document de politique JSON suivant :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
```

```
            "Resource": ["arn:aws:logs:*:*:*"]
        }
    ]
}
```

6. Choisissez Suivant.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez la page Restructuration de politique dans le Guide de l'utilisateur IAM.

7. Sur la page Vérifier et créer, saisissez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique.
8. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Après avoir créé la politique, attachez-la au rôle d'instance de conteneur. Pour plus d'informations sur la manière d'associer la politique au rôle, consultez la section Mise à jour des autorisations pour un rôle dans le Guide de AWS Identity and Access Management l'utilisateur.

AWS CLI

1. Créez un fichier nommé `instance-cw-logs.json` avec le contenu suivant.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
```

```
            ],
            "Resource": ["arn:aws:logs:*:*:*"]
        }
    ]
}
```

2. Utilisez la commande suivante pour créer la stratégie IAM à l'aide du fichier de document de stratégie JSON.

```
aws iam create-policy \
      --policy-name cwlogspolicy \
      --policy-document file://instance-cw-logs.json
```

3. Récupérez l'ARN de la politique IAM que vous avez créée à l'aide de la commande suivante. *cwlogspolicy*Remplacez-le par le nom de la politique que vous avez créée.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`cwlogspolicy`].Arn'
```

4. Utilisez la commande suivante pour associer la politique au rôle IAM de l'instance de conteneur à l'aide de l'ARN de la stratégie.

```
aws iam attach-role-policy \
      --role-name ecsInstanceRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/cwlogspolicy
```

Rôle IAM dans Amazon ECS Anywhere

Lorsque vous enregistrez un serveur local ou une machine virtuelle (VM) dans votre cluster, le serveur ou la machine virtuelle a besoin d'un rôle IAM pour communiquer avec celui-ci. AWS APIs Vous ne devez créer ce rôle IAM qu'une seule fois pour chaque AWS compte. Toutefois, ce rôle IAM doit être associé à chaque serveur ou machine virtuelle que vous enregistrez sur un cluster. Ce rôle est le `ECSAnywhereRole`. Vous pouvez créer ce rôle manuellement. De même, Amazon ECS peut créer le rôle en votre nom lorsque vous enregistrez une instance externe dans la AWS Management Console. Vous pouvez utiliser la recherche dans la console IAM pour rechercher `ecsAnywhereRole` et vérifier si votre compte possède déjà le rôle. Pour plus d'informations, consultez la section Recherche dans la console IAM dans le guide de l'utilisateur d'IAM.

AWS fournit deux politiques IAM gérées qui peuvent être utilisées lors de la création du rôle IAM ECS Anywhere, à savoir les politiques

`AmazonSSMManagedInstanceCore` et`AmazonEC2ContainerServiceforEC2Role`. La stratégie `AmazonEC2ContainerServiceforEC2Role` inclut des autorisations qui fournissent probablement plus d'accès que vous n'avez besoin. Par conséquent, en fonction de votre cas d'utilisation spécifique, nous vous recommandons de créer une stratégie personnalisée en ajoutant uniquement les autorisations de cette stratégie dont vous avez besoin. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.

Le rôle IAM d'exécution de tâche qui accorde à l'agent de conteneur Amazon ECS l'autorisation d'effectuer des appels d'API AWS en votre nom. Lorsqu'un rôle IAM d'exécution de tâche est utilisé, il doit être spécifié dans votre définition de tâche. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

Le rôle d'exécution de tâche est requis si l'une des conditions suivantes s'applique :

- Vous envoyez des journaux de conteneurs à CloudWatch Logs à l'aide du pilote de `awslogs` journal.
- Votre définition de tâche spécifie une image de conteneur hébergée dans un référentiel privé Amazon ECR. Toutefois, si le `ECSAnywhereRole` rôle associé à votre instance externe inclut également les autorisations nécessaires pour extraire des images d'Amazon ECR, votre rôle d'exécution de tâches n'a pas besoin de les inclure.

Création du rôle Amazon ECS Anywhere

Remplacez tout *`user input`* par vos propres informations.

1. Créez un fichier local nommé `ssm-trust-policy.json` avec la politique de confiance suivante.

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Principal": {"Service": [
      "ssm.amazonaws.com"
    ]},
    "Action": "sts:AssumeRole"
  }
```

```
}
```

2. Créez le rôle et associez la politique de confiance à l'aide de la AWS CLI commande suivante.

```
aws iam create-role --role-name ecsAnywhereRole --assume-role-policy-document
 file://ssm-trust-policy.json
```

3. Joignez les politiques AWS gérées à l'aide de la commande suivante.

```
aws iam attach-role-policy --role-name ecsAnywhereRole --policy-arn
 arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
aws iam attach-role-policy --role-name ecsAnywhereRole --policy-arn
 arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role
```

Vous pouvez également utiliser le flux de travail de politique de confiance personnalisé IAM pour créer le rôle. Pour plus d'informations, consultez la section Création d'un rôle à l'aide de politiques de confiance personnalisées (console) dans le guide de l'utilisateur IAM.

Rôle IAM dans l'infrastructure Amazon ECS

Un rôle IAM dans l'infrastructure Amazon ECS permet à Amazon ECS de gérer les ressources d'infrastructure de vos clusters en votre nom. Il est utilisé lorsque :

- Vous souhaitez associer des volumes Amazon EBS à vos tâches Amazon ECS de type Fargate ou de EC2 lancement. Le rôle d'infrastructure permet à Amazon ECS de gérer les volumes Amazon EBS pour vos tâches.
- Vous souhaitez utiliser le protocole TLS (Transport Layer Security) pour chiffrer le trafic entre vos services Amazon ECS Service Connect.
- Vous souhaitez créer des groupes cibles Amazon VPC Lattice.

Lorsqu'Amazon ECS assume ce rôle pour prendre des mesures en votre nom, les événements seront visibles dans AWS CloudTrail. Si Amazon ECS utilise le rôle pour gérer les volumes Amazon EBS associés à vos tâches, le CloudTrail journal le `roleSessionName` sera`ECSTaskVolumesForEBS`. Si le rôle est utilisé pour chiffrer le trafic entre vos services Service Connect, le CloudTrail journal le `roleSessionName` sera`ECSServiceConnectForTLS`. Si le rôle est utilisé pour créer des groupes cibles pour VPC Lattice, le CloudTrail journal `roleSessionName` le sera. `ECSNetworkingWithVPCLattice` Vous pouvez utiliser ce nom pour rechercher des événements dans la CloudTrail console en filtrant par nom d'utilisateur.

Amazon ECS fournit des politiques gérées qui contiennent les autorisations requises pour l'attachement de volumes et le protocole TLS. Pour plus d'informations, consultez Amazon ECSInfrastructure RolePolicyForVolumes ECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity, Amazon et Amazon ECSInfrastructure RolePolicyForVpcLattice dans le AWS Managed Policy Reference Guide.

Création du rôle d'infrastructure Amazon ECS

Remplacez tout *`user input`* par vos propres informations.

1. Créez un fichier nommé `ecs-infrastructure-trust-policy.json` contenant la stratégie d'approbation à utiliser pour le rôle IAM. Le fichier doit contenir ce qui suit :

   JSON

   ```
   {
     "Version": "2012-10-17",
     "Statement": [
       {
         "Sid": "AllowAccessToECSForInfrastructureManagement",
         "Effect": "Allow",
         "Principal": {
           "Service": "ecs.amazonaws.com"
         },
         "Action": "sts:AssumeRole"
       }
     ]
   }
   ```

2. Utilisez la AWS CLI commande suivante pour créer un rôle nommé `ecsInfrastructureRole` en utilisant la politique de confiance que vous avez créée à l'étape précédente.

   ```
   aws iam create-role \
         --role-name ecsInfrastructureRole \
         --assume-role-policy-document file://ecs-infrastructure-trust-policy.json
   ```

3. En fonction de votre cas d'utilisation, associez le rôle AWS géré `AmazonECSInfrastructureRolePolicyForVolumes` ou la `AmazonECSInfrastructureRolePolicyForVpcLattice` politique au `ecsInfrastructureRole` rôle. `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity`

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSInfrastructureRolePolicyForVolumes
```

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity
```

Vous pouvez également utiliser le flux de travail de politique de confiance personnalisée de la console IAM pour créer le rôle. Pour plus d'informations, consultez la section Création d'un rôle à l'aide de politiques de confiance personnalisées (console) dans le guide de l'utilisateur IAM.

Important

Si le rôle d'infrastructure ECS est utilisé par Amazon ECS pour gérer les volumes Amazon EBS associés à vos tâches, assurez-vous de ce qui suit avant d'arrêter les tâches qui utilisent des volumes Amazon EBS.

- Le rôle n'est pas supprimé.
- La politique de confiance associée au rôle n'est pas modifiée pour supprimer l'accès Amazon ECS (`ecs.amazonaws.com`).
- La politique gérée `AmazonECSInfrastructureRolePolicyForVolumes` n'est pas supprimée. Si vous devez modifier les autorisations du rôle, conservez-les au moins `ec2:DetachVolumeec2:DeleteVolume`, et `ec2:DescribeVolumes` pour la suppression du volume.

Si vous supprimez ou modifiez le rôle avant d'arrêter les tâches associées à des volumes Amazon EBS, les tâches resteront bloquées `DEPROVISIONING` et les volumes Amazon EBS associés ne seront pas supprimés. Amazon ECS réessaiera automatiquement à intervalles réguliers d'arrêter la tâche et de supprimer le volume jusqu'à ce que les autorisations nécessaires soient rétablies. Vous pouvez consulter l'état d'attachement au volume d'une tâche et la raison du statut associée à l'aide de l'DescribeTasksAPI.

Après avoir créé le fichier, vous devez accorder à votre utilisateur l'autorisation de transmettre le rôle à Amazon ECS.

Autorisation de transmettre le rôle d'infrastructure à Amazon ECS

Pour utiliser un rôle IAM dans l'infrastructure ECS, vous devez autoriser votre utilisateur à transmettre le rôle à Amazon ECS. Attachez l'`iam:PassRole`autorisation suivante à votre utilisateur. *`ecsInfrastructureRole`*Remplacez-le par le nom du rôle d'infrastructure que vous avez créé.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [

        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource": ["arn:aws:iam::*:role/ecsInfrastructureRole"],
            "Condition": {
                "StringEquals": {"iam:PassedToService": "ecs.amazonaws.com"}
            }
        }
    ]
}
```

Pour plus d'informations sur les autorisations de votre utilisateur `iam:Passrole` et leur mise à jour, consultez les sections Octroi à un utilisateur des autorisations lui permettant de transférer un rôle à un AWS service et Modification des autorisations d'un utilisateur IAM dans le guide de l'AWS Identity and Access Management utilisateur.

Rôle IAM de l'infrastructure Amazon ECS pour les équilibreurs de charge

Un rôle IAM dans l'infrastructure Amazon ECS pour les équilibreurs de charge permet à Amazon ECS de gérer les ressources des équilibreurs de charge dans vos clusters en votre nom. Il est utilisé lorsque :

- Vous souhaitez utiliser des blue/green déploiements avec Amazon ECS. Le rôle d'infrastructure permet à Amazon ECS de gérer les ressources de l'équilibreur de charge pour vos déploiements.

- Vous avez besoin d'Amazon ECS pour créer, modifier ou supprimer des ressources d'équilibrage de charge telles que des groupes cibles et des écouteurs lors des opérations de déploiement.

Lorsqu'Amazon ECS assume ce rôle pour prendre des mesures en votre nom, les événements seront visibles dans AWS CloudTrail. Si Amazon ECS utilise le rôle pour gérer les ressources de l'équilibreur de charge pour vos déploiements bleu/vert, le CloudTrail journal `roleSessionName` sera ou. `ECSNetworkingWithELB` `ecs-service-scheduler` Vous pouvez utiliser ce nom pour rechercher des événements dans la CloudTrail console en filtrant par nom d'utilisateur.

Amazon ECS fournit une politique gérée qui contient les autorisations requises pour la gestion de l'équilibreur de charge. Pour plus d'informations, consultez Amazon ECSInfrastructure RolePolicyForLoadBalancers dans le AWS Managed Policy Reference Guide.

Création du rôle d'infrastructure Amazon ECS pour les équilibreurs de charge

Remplacez tout *`user input`* par vos propres informations.

1. Créez un fichier nommé `ecs-infrastructure-trust-policy.json` contenant la stratégie d'approbation à utiliser pour le rôle IAM. Le fichier doit contenir ce qui suit :

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToECSForInfrastructureManagement",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Utilisez la AWS CLI commande suivante pour créer un rôle nommé `ecsInfrastructureRoleForLoadBalancers` en utilisant la politique de confiance que vous avez créée à l'étape précédente.

```
aws iam create-role \
      --role-name ecsInfrastructureRoleForLoadBalancers \
      --assume-role-policy-document file://ecs-infrastructure-trust-policy.json
```

3. Associez la `AmazonECSInfrastructureRolePolicyForLoadBalancers` politique AWS gérée au `ecsInfrastructureRoleForLoadBalancers` rôle.

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRoleForLoadBalancers \
      --policy-arn arn:aws:iam::aws:policy/
AmazonECSInfrastructureRolePolicyForLoadBalancers
```

Vous pouvez également utiliser le flux de travail de politique de confiance personnalisée de la console IAM pour créer le rôle. Pour plus d'informations, consultez la section Création d'un rôle à l'aide de politiques de confiance personnalisées (console) dans le guide de l'utilisateur IAM.

> **Important**
>
> Si le rôle d'infrastructure est utilisé par Amazon ECS pour gérer les ressources de l'équilibreur de charge pour vos blue/green déploiements, assurez-vous de ce qui suit avant de supprimer ou de modifier le rôle :
>
> - Le rôle n'est pas supprimé lorsque les déploiements actifs sont en cours.
> - La politique de confiance associée au rôle n'est pas modifiée pour supprimer l'accès Amazon ECS (`ecs.amazonaws.com`).
> - La politique gérée `AmazonECSInfrastructureRolePolicyForLoadBalancers` n'est pas supprimée lorsque les déploiements actifs sont en cours.
>
> La suppression ou la modification du rôle pendant blue/green les déploiements actifs peut entraîner des échecs de déploiement et laisser vos services dans un état incohérent.

Après avoir créé le fichier, vous devez accorder à votre utilisateur l'autorisation de transmettre le rôle à Amazon ECS.

Autorisation de transmettre le rôle d'infrastructure à Amazon ECS

Pour utiliser un rôle IAM d'infrastructure ECS pour les équilibreurs de charge, vous devez accorder à votre utilisateur l'autorisation de transmettre le rôle à Amazon ECS. Attachez l'`iam:PassRole`autorisation suivante à votre utilisateur. *`ecsInfrastructureRoleForLoadBalancers`*Remplacez-le par le nom du rôle d'infrastructure que vous avez créé.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource":
 ["arn:aws:iam::*:role/ecsInfrastructureRoleForLoadBalancers"],
            "Condition": {
                "StringEquals": {"iam:PassedToService": "ecs.amazonaws.com"}
            }
        }
    ]
}
```

Pour plus d'informations sur les autorisations de votre utilisateur `iam:Passrole` et leur mise à jour, consultez les sections Octroi à un utilisateur des autorisations lui permettant de transférer un rôle à un AWS service et Modification des autorisations d'un utilisateur IAM dans le guide de l'AWS Identity and Access Management utilisateur.

Rôle CodeDeploy IAM d'Amazon ECS

Avant de pouvoir utiliser le type de déploiement CodeDeploy bleu/vert avec Amazon ECS, le CodeDeploy service a besoin d'autorisations pour mettre à jour votre service Amazon ECS en votre nom. Ces autorisations sont fournies par le rôle CodeDeploy IAM (`ecsCodeDeployRole`).

> **Note**
>
> Les utilisateurs ont également besoin d'autorisations pour les utiliser CodeDeploy ; ces autorisations sont décrites dansAutorisations IAM requises.

Deux politiques gérées sont fournies. Pour plus d'informations, consultez l'un des articles suivants dans le Guide de référence des politiques AWS gérées :

- AWSCodeDeployRoleForECS - donne CodeDeploy l'autorisation de mettre à jour n'importe quelle ressource à l'aide de l'action associée.
- AWSCodeDeployRoleForECSLimited- donne des autorisations CodeDeploy plus limitées.

Création du CodeDeploy rôle

Vous pouvez utiliser les procédures suivantes pour créer un CodeDeploy rôle pour Amazon ECS

AWS Management Console

Pour créer le rôle de service pour CodeDeploy (console IAM)

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le volet de navigation de la console IAM, sélectionnez Roles (Rôles), puis Create role (Créer un rôle).
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez CodeDeploy, puis choisissez le cas d'utilisation CodeDeploy - ECS.
5. Choisissez Suivant.
6. Dans la section Attacher une politique d'autorisations, assurez-vous que la stratégie AWSCodeDeployRoleForECS est sélectionnée.
7. Choisissez Suivant.
8. Dans Nom du rôle, entrez ecsCodeDeployRôle.
9. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

AWS CLI

Remplacez tout *user input* par vos propres informations.

1. Créez un fichier nommé `codedeploy-trust-policy.json` contenant la politique de confiance à utiliser pour le rôle CodeDeploy IAM.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": ["codedeploy.amazonaws.com"]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

2. Créez un rôle IAM nommé `ecsCodedeployRole` à l'aide de la stratégie d'approbation créée à l'étape précédente.

```
aws iam create-role \
      --role-name ecsCodedeployRole \
      --assume-role-policy-document file://codedeploy-trust-policy.json
```

3. Attachez la politique `AWSCodeDeployRoleForECS` ou la politique `AWSCodeDeployRoleForECSLimited` gérée au `ecsTaskRole` rôle.

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam::aws:policy/AWSCodeDeployRoleForECS
```

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam::aws:policy/AWSCodeDeployRoleForECSLimited
```

Lorsque les tâches de votre service nécessitent un rôle d'exécution de tâches, vous devez ajouter l'`iam:PassRole`autorisation pour chaque rôle d'exécution de tâche ou chaque remplacement de rôle de tâche au CodeDeploy rôle en tant que politique.

Autorisations relatives aux rôles d'exécution des tâches

Lorsque les tâches de votre service nécessitent un rôle d'exécution de tâches, vous devez ajouter l'`iam:PassRole`autorisation pour chaque rôle d'exécution de tâche ou chaque remplacement de rôle de tâche au CodeDeploy rôle en tant que politique. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS et Rôle IAM de la tâche Amazon ECS. Ensuite, vous associez cette politique au CodeDeploy rôle

Création de la stratégie

AWS Management Console

Pour utiliser l'éditeur de politique JSON afin de créer une politique

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).

   Si vous sélectionnez Politiques pour la première fois, la page Bienvenue dans les politiques gérées s'affiche. Sélectionnez Mise en route.
3. En haut de la page, sélectionnez Créer une politique.
4. Dans la section Éditeur de politique, choisissez l'option JSON.
5. Entrez le document de politique JSON suivant :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsCodeDeployRole>"]
        }
    ]
}
```

6. Choisissez Suivant.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez la page Restructuration de politique dans le Guide de l'utilisateur IAM.

7. Sur la page Vérifier et créer, saisissez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique.
8. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Après avoir créé la stratégie, associez-la au CodeDeploy rôle. Pour plus d'informations sur la manière d'associer la politique au rôle, voir Mettre à jour les autorisations pour un rôle dans le Guide de AWS Identity and Access Management l'utilisateur.

AWS CLI

Remplacez tout *`user input`* par vos propres informations.

1. Créez un fichier nommé `blue-green-iam-passrole.json` avec le contenu suivant.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                   "arn:arn:aws:iam::111122223333:role/ecs-code-deploy-
role"
            ]
        }
    ]
}
```

2. Utilisez la commande suivante pour créer la stratégie IAM à l'aide du fichier de document de stratégie JSON.

```
aws iam create-policy \
      --policy-name cdTaskExecutionPolicy \
      --policy-document file://blue-green-iam-passrole.json
```

3. Récupérez l'ARN de la politique IAM que vous avez créée à l'aide de la commande suivante.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`cdTaskExecutionPolicy`].Arn'
```

4. Utilisez la commande suivante pour associer la politique au rôle CodeDeploy IAM.

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/cdTaskExecutionPolicy
```

Rôle EventBridge IAM d'Amazon ECS

Avant de pouvoir utiliser les tâches planifiées Amazon ECS avec des EventBridge règles et des cibles, le EventBridge service a besoin d'autorisations pour exécuter des tâches Amazon ECS en votre nom. Ces autorisations sont fournies par le rôle IAM EventBridge (`ecsEventsRole`).

La politique `AmazonEC2ContainerServiceEventsRole` est présentée ci-dessous.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:RunTask"],
            "Resource": ["*"]
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["*"],
```

```
            "Condition": {
                "StringLike": {"iam:PassedToService": "ecs-tasks.amazonaws.com"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "ecs:TagResource",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:CreateAction": ["RunTask"]
                }
            }
        }
    ]
}
```

Si vos tâches planifiées nécessitent l'utilisation du rôle d'exécution de tâches, d'un rôle de tâche ou d'un remplacement de rôle de tâche, vous devez ajouter des `iam:PassRole` autorisations pour chaque rôle d'exécution de tâche, rôle de tâche ou remplacement de rôle de tâche au rôle EventBridge IAM. Pour plus d'informations sur le rôle d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS.

> **Note**
>
> Spécifiez l'ARN complet du rôle d'exécution de tâche ou du remplacement de rôle de tâche.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [

 "arn:aws:iam::111122223333:role/ecsTaskExecutionRole_or_TaskRole_name"
            ]
```

```
        }
    ]
}
```

Vous pouvez choisir de le laisser AWS Management Console créer le EventBridge rôle pour vous lorsque vous configurez une tâche planifiée. Pour de plus amples informations, veuillez consulter Utilisation d'Amazon EventBridge Scheduler pour planifier des tâches Amazon ECS .

Création du EventBridge rôle

Remplacez tout *`user input`* par vos propres informations.

1. Créez un fichier nommé `eventbridge-trust-policy.json` contenant la stratégie d'approbation à utiliser pour le rôle IAM. Le fichier doit contenir ce qui suit :

   JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Utilisez la commande suivante pour créer un rôle IAM nommé `ecsEventsRole` en utilisant la politique de confiance que vous avez créée à l'étape précédente.

```
aws iam create-role \
      --role-name ecsEventsRole \
      --assume-role-policy-document file://eventbridge-trust-policy.json
```

3. Attachez le AWS `AmazonEC2ContainerServiceEventsRole` managé au `ecsEventsRole` rôle à l'aide de la commande suivante.

```
aws iam attach-role-policy \
      --role-name ecsEventsRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceEventsRole
```

Vous pouvez également utiliser le flux de travail de politique de confiance personnalisé (https://console.aws.amazon.com/iam/) de la console IAM pour créer le rôle. Pour plus d'informations, consultez la section Création d'un rôle à l'aide de politiques de confiance personnalisées (console) dans le guide de l'utilisateur IAM.

Attachement d'une stratégie au rôle **`ecsEventsRole`**

Vous pouvez utiliser les procédures suivantes pour ajouter des autorisations pour le rôle d'exécution de tâches au rôle EventBridge IAM.

AWS Management Console

Pour utiliser l'éditeur de politique JSON afin de créer une politique

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.
2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).

   Si vous sélectionnez Politiques pour la première fois, la page Bienvenue dans les politiques gérées s'affiche. Sélectionnez Mise en route.
3. En haut de la page, sélectionnez Créer une politique.
4. Dans la section Éditeur de politique, choisissez l'option JSON.
5. Entrez le document de politique JSON suivant :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsTaskExecutionRole_or_TaskRole_name>"]
        }
```

```
    ]
}
```

6. Choisissez Suivant.

> **Note**
>
> Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez la page Restructuration de politique dans le Guide de l'utilisateur IAM.

7. Sur la page Vérifier et créer, saisissez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique.
8. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Après avoir créé la stratégie, associez-la au EventBridge rôle. Pour plus d'informations sur la manière d'associer la politique au rôle, voir Mettre à jour les autorisations pour un rôle dans le Guide de AWS Identity and Access Management l'utilisateur.

AWS CLI

Remplacez tout *`user input`* par vos propres informations.

1. Créez un fichier nommé `ev-iam-passrole.json` avec le contenu suivant.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [

 "arn:aws:iam::111122223333:role/ecsTaskExecutionRole_or_TaskRole_name"
            ]
```

```
        }
    ]
}
```

2. Utilisez la AWS CLI commande suivante pour créer la stratégie IAM à l'aide du fichier de document de stratégie JSON.

```
aws iam create-policy \
      --policy-name eventsTaskExecutionPolicy \
      --policy-document file://ev-iam-passrole.json
```

3. Récupérez l'ARN de la politique IAM que vous avez créée à l'aide de la commande suivante.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`eventsTaskExecutionPolicy`].Arn'
```

4. Utilisez la commande suivante pour associer la stratégie au rôle EventBridge IAM à l'aide de l'ARN de la stratégie.

```
aws iam attach-role-policy \
      --role-name ecsEventsRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/eventsTaskExecutionPolicy
```

Autorisations requises pour la console Amazon ECS

Conformément à la bonne pratique pour accorder le moindre privilège, vous pouvez utiliser la stratégie gérée par `AmazonECS_FullAccess` comme modèle pour créer votre propre stratégie personnalisée. De cette façon, vous pouvez supprimer ou ajouter des autorisations pour la stratégie gérée en fonction de vos besoins spécifiques. Pour plus d'informations, consultez AmazonECS_FullAccess dans le manuel AWS Managed Policy Reference.

Autorisations pour créer des rôles IAM

Les actions suivantes nécessitent des autorisations supplémentaires pour terminer l'opération :

- Enregistrement d'une instance externe : pour plus d'informations, veuillez consulter Rôle IAM dans Amazon ECS Anywhere.
- Enregistrement d'une définition de tâche : pour plus d'informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

- Création d'une EventBridge règle à utiliser pour planifier des tâches - pour plus d'informations, voir Rôle EventBridge IAM d'Amazon ECS

Vous pouvez ajouter ces autorisations en créant un rôle dans IAM avant de les utiliser dans la console Amazon ECS. Si vous ne créez pas les rôles, la console Amazon ECS les crée en votre nom.

Autorisations requises pour enregistrer une instance externe dans un cluster

Vous avez besoin d'autorisations supplémentaires lorsque vous enregistrez une instance externe dans un cluster et que vous souhaitez créer un nouveau rôle d'instance externe (`ecsExternalInstanceRole`).

Les autorisations supplémentaires suivantes sont requises :

- `iam` : permet aux principaux de créer et répertorier les rôles IAM et les politiques qui leur sont attachées.
- `ssm` : permet aux principaux d'enregistrer l'instance externe auprès de Systems Manager.

> **Note**
>
> Pour choisir un `ecsExternalInstanceRole` existant, vous devez disposer des autorisations `iam:GetRole` et `iam:PassRole`.

La politique suivante contient les autorisations requises et limite les actions au rôle `ecsExternalInstanceRole`.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:CreateInstanceProfile",
              "iam:AddRoleToInstanceProfile",
              "iam:ListInstanceProfilesForRole",
              "iam:GetRole"
```

```
            ],
            "Resource": "arn:aws:iam::*:role/ecsExternalInstanceRole"
        },
        {
            "Effect": "Allow",
            "Action": ["iam:PassRole","ssm:CreateActivation"],
            "Resource": "arn:aws:iam::*:role/ecsExternalInstanceRole"
        }
    ]
}
```

Autorisations requises pour enregistrer une définition de tâche

Vous avez besoin d'autorisations supplémentaires lorsque vous enregistrez une définition de tâche et que vous souhaitez créer un nouveau rôle d'exécution de tâche (`ecsTaskExecutionRole`).

Les autorisations supplémentaires suivantes sont requises :

- `iam` : permet aux principaux de créer et répertorier les rôles IAM et les politiques qui leur sont attachées.

> **Note**
>
> Pour choisir un `ecsTaskExecutionRole` existant, vous devez disposer de l'autorisation `iam:GetRole`.

La politique suivante contient les autorisations requises et limite les actions au rôle `ecsTaskExecutionRole`.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsTaskExecutionRole"
```

```
        }
    ]
}
```

Autorisations requises pour créer une EventBridge règle pour les tâches planifiées

Vous avez besoin d'autorisations supplémentaires lorsque vous planifiez une tâche et que vous souhaitez créer un nouveau rôle CloudWatch Events (`ecsEventsRole`).

Les autorisations supplémentaires suivantes sont requises :

- `iam` : permet aux principaux de créer et de répertorier les rôles IAM et les politiques qui leur sont associées, et d'autoriser Amazon ECS à transmettre le rôle à d'autres services pour qu'ils l'endossent.

> **Note**
>
> Pour choisir un `ecsEventsRole` existant, vous devez disposer des autorisations `iam:GetRole` et `iam:PassRole`.

La politique suivante contient les autorisations requises et limite les actions au rôle `ecsEventsRole`.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:GetRole",
              "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsEventsRole"
        }
    ]
}
```

Autorisations requises pour visualiser les déploiements de services

Lorsque vous suivez la meilleure pratique consistant à accorder le moindre privilège, vous devez ajouter des autorisations supplémentaires afin de visualiser les déploiements de services dans la console.

Vous devez avoir accès aux actions suivantes :

- ListServiceDeployments
- DescribeServiceDeployments
- DescribeServiceRevisions

Vous devez avoir accès aux ressources suivantes :

- Service
- Déploiement du service
- Révision du service

L'exemple de politique suivant contient les autorisations requises et limite les actions à un service spécifique.

Remplacez le `accountcluster-name`, et `service-name` par vos valeurs.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "ecs:ListServiceDeployments",
              "ecs:DescribeServiceDeployments",
              "ecs:DescribeServiceRevisions"
            ],
            "Resource": [
             "arn:aws:ecs:us-east-1:123456789012:service/cluster-name/service-name",
             "arn:aws:ecs:us-east-1:123456789012:service-deployment/cluster-name/
service-name/*",
             "arn:aws:ecs:us-east-1:123456789012:service-revision/cluster-name/service-
name/*"
            ]
```

```
        }
    ]
}
```

Autorisations requises pour afficher les événements du cycle de vie Amazon ECS dans Container Insights

Les autorisations suivantes sont requises pour afficher les événements du cycle de vie. Ajoutez les autorisations suivantes en tant que politique intégrée au rôle. Pour plus d'informations, consultez Ajout et suppression de politiques IAM.

- événements : DescribeRule
- événements : ListTargetsByRule
- journaux : DescribeLogGroups

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:DescribeRule",
        "events:ListTargetsByRule",
        "logs:DescribeLogGroups"
      ],
      "Resource": "*"
    }
  ]
}
```

Autorisations requises pour activer les événements du cycle de vie d'Amazon ECS dans Container Insights

Les autorisations suivantes sont requises pour configurer les événements du cycle de vie :

- événements : PutRule

- événements : PutTargets
- journaux : CreateLogGroup

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutRule",
        "events:PutTargets",
        "logs:CreateLogGroup"
      ],
      "Resource": "*"
    }
  ]
}
```

Autorisations requises pour la console Amazon ECS avec AWS CloudFormation

Avant d'utiliser le AWS Management Console pour créer vos ressources, vous devez vous assurer que vous disposez des autorisations IAM appropriées. Pour plus d'informations sur la façon de configurer les autorisations pour la console Amazon ECS en général, consultez d'abord la section Autorisations requises pour la console Amazon ECS.

La console Amazon ECS est alimentée par AWS CloudFormation et nécessite des autorisations IAM supplémentaires dans les cas suivants :

- Création d’un cluster
- Création d'un service
- Création d'un fournisseur de capacité

Vous pouvez créer une politique pour les autorisations supplémentaires, puis les associer au rôle IAM que vous utilisez pour accéder à la console. Pour plus d’informations, consultez Création de politiques IAM dans le Guide de l’utilisateur IAM.

Autorisations requises pour créer un cluster

Lorsque vous créez un cluster dans la console, vous avez besoin d'autorisations supplémentaires qui vous permettent de gérer les AWS CloudFormation piles.

Les autorisations supplémentaires suivantes sont requises :

- `cloudformation` : permet aux mandataires de créer et de gérer des piles AWS CloudFormation . Ceci est requis lors de la création de clusters Amazon ECS à l'aide de la AWS Management Console et pour la gestion ultérieure de ces groupes.
- `ssm`— Permet de AWS CloudFormation référencer la dernière AMI optimisée pour Amazon ECS. Cela est nécessaire lors de la création de clusters Amazon ECS à l'aide du AWS Management Console.

La politique suivante contient les AWS CloudFormation autorisations requises et limite les actions aux ressources créées dans la console Amazon ECS.

```
{
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack"
             ],
            "Resource": [
                "arn:*:cloudformation:*:*:stack/Infra-ECS-Cluster-*"
            ]
      },
      {
          "Effect": "Allow",
          "Action": "ssm:GetParameters",
          "Resource": [
            "arn:aws:ssm:*:*:parameter/aws/service/ecs/optimized-ami/amazon-linux-2*/
*",
            "arn:aws:ssm:*:*:parameter/aws/service/ecs/optimized-ami/amazon-
linux-2023*/*"
          ]
      }
```

```
    ]
}
```

Si vous n'avez pas créé le rôle d'instance de conteneur Amazon ECS (`ecsInstanceRole`) et que vous créez un cluster qui utilise des EC2 instances Amazon, la console créera le rôle en votre nom.

En outre, si vous utilisez des groupes Auto Scaling, vous avez besoin d'autorisations supplémentaires pour que la console puisse ajouter des balises aux groupes de mise à l'échelle automatique lorsque vous utilisez la fonctionnalité de mise à l'échelle automatique du cluster.

Les autorisations supplémentaires suivantes sont requises :

- `autoscaling`— Permet à la console de baliser le groupe Amazon EC2 Auto Scaling. Cela est nécessaire lors de la gestion des groupes Amazon EC2 Auto Scaling lors de l'utilisation de la fonctionnalité de dimensionnement automatique des clusters. La balise est la balise gérée par ECS que la console ajoute automatiquement au groupe pour indiquer qu'il a été créé dans la console.
- `iam` : permet aux mandataires de répertorier les rôles IAM et les stratégies qui leur sont attachées. Les principaux peuvent également répertorier les profils d'instance disponibles pour vos EC2 instances Amazon.

La politique suivante contient les autorisations IAM requises et limite les actions au rôle `ecsInstanceRole`.

Les autorisations Auto Scaling ne sont pas limitées.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:CreateInstanceProfile",
              "iam:AddRoleToInstanceProfile",
              "iam:ListInstanceProfilesForRole",
              "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsInstanceRole"
        },
        {
```

```
            "Effect": "Allow",
            "Action": "autoscaling:CreateOrUpdateTags",
            "Resource": "*"
        }
    ]
}
```

Autorisations requises pour créer un service

Lorsque vous créez un service dans la console, vous avez besoin d'autorisations supplémentaires qui vous permettent de gérer les AWS CloudFormation piles. Les autorisations supplémentaires suivantes sont requises :

- `cloudformation` : permet aux mandataires de créer et de gérer des piles AWS CloudFormation . Cela est requis lors de la création de services Amazon ECS à l'aide de l' AWS Management Console et pour la gestion ultérieure de ces groupes.

La politique suivante contient les autorisations requises et limite les actions aux ressources créées dans la console Amazon ECS.

```
{
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack"
             ],
            "Resource": [
                "arn:*:cloudformation:*:*:stack/ECS-Console-V2-Service-*"
            ]
      }
   ]
}
```

Autorisations requises pour les fonctions Lambda dans les déploiements Amazon ECS blue/green

Lorsque vous utilisez les fonctions Lambda comme points d'ancrage du cycle de vie de déploiement dans les déploiements bleu/vert d'Amazon ECS, vous devez créer un rôle IAM avec des autorisations spécifiques. Ce rôle permet à Amazon ECS d'appeler vos fonctions Lambda à différentes étapes du cycle de vie du déploiement.

Les autorisations supplémentaires suivantes sont requises :

- `lambda:InvokeFunction`— Permet à Amazon ECS d'invoquer des fonctions Lambda configurées comme des hooks du cycle de vie pendant le processus de déploiement.

La politique suivante contient les autorisations requises pour les hooks du cycle de vie Lambda :

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "lambda:InvokeFunction"
            ],
            "Resource": "arn:aws:lambda:*:*:function:*"
        }
    ]
}
```

Pour la politique de confiance, vous devez autoriser le service à assumer ce rôle :

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ecs.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Autorisations IAM requises pour le dimensionnement automatique du service Amazon ECS

Service Auto Scaling est rendu possible par la combinaison d'Amazon ECS et d'Application Auto Scaling APIs. CloudWatch Les services sont créés et mis à jour avec Amazon ECS, les alarmes sont créées avec CloudWatch Application Auto Scaling et les politiques de dimensionnement sont créées avec Application Auto Scaling.

Outre les autorisations IAM standard pour créer et mettre à jour des services, les autorisations suivantes sont requises pour interagir avec les paramètres de Service Auto Scaling, comme indiqué dans l'exemple de politique suivant.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:*",
                "ecs:DescribeServices",
                "ecs:UpdateService",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarmHistory",
                "cloudwatch:DescribeAlarmsForMetric",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics",
                "cloudwatch:DisableAlarmActions",
                "cloudwatch:EnableAlarmActions",
```

```
                "iam:CreateServiceLinkedRole",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Get*",
                "sns:List*"
            ],
            "Resource": ["*"]
        }
    ]
}
```

Les exemples de politique IAM Créer un exemple de service Amazon ECS et Exemple de mise à jour du service Amazon ECS présentent les autorisations nécessaires pour utiliser Service Auto Scaling dans la AWS Management Console.

Le service Application Auto Scaling a également besoin d'une autorisation pour décrire vos services et CloudWatch alarmes Amazon ECS, ainsi que d'autorisations pour modifier le nombre souhaité pour votre service en votre nom. Les `sns:` autorisations concernent les notifications CloudWatch envoyées à une rubrique Amazon SNS lorsqu'un seuil est dépassé. Si vous utilisez la scalabilité automatique pour vos services Amazon ECS services, elle crée un rôle lié à un service nommé `AWSServiceRoleForApplicationAutoScaling_ECSService`. Ce rôle lié à un service accorde l'autorisation Application Auto Scaling de décrire les alarmes de vos politiques, de contrôler le nombre de tâches en cours d'exécution du service et de modifier le nombre souhaité du service. Le rôle d'origine géré par Amazon ECS pour Application Auto Scaling était `ecsAutoscaleRole`, mais il n'est plus nécessaire. Le rôle lié à un service est le rôle par défaut pour Application Auto Scaling. Pour plus d'informations, consultez Rôles liés aux services pour Application Auto Scaling dans le Guide de l'utilisateur Application Auto Scaling.

Si vous avez créé votre rôle d'instance de conteneur Amazon ECS avant que les CloudWatch métriques ne soient disponibles pour Amazon ECS, vous devrez peut-être ajouter l'`ecs:StartTelemetrySession`autorisation. Pour de plus amples informations, veuillez consulter Considérations.

Octroi de l'autorisation de baliser les ressources lors de la création

Les actions de balisage lors de la création d'API Amazon ECS suivantes vous permettent de spécifier des balises lorsque vous créez la ressource. Si des balises sont spécifiées dans l'action de création de ressources, AWS effectue une autorisation supplémentaire pour vérifier que les autorisations appropriées sont attribuées pour créer des balises.

- `CreateCapacityProvider`
- `CreateCluster`
- `CreateService`
- `CreateTaskSet`
- `RegisterContainerInstance`
- `RegisterTaskDefinition`
- `RunTask`
- `StartTask`

Vous pouvez utiliser des balises de ressource pour implémenter le contrôle basé sur les attributs (ABAC). Pour plus d'informations, consultez the section called "Contrôle de l'accès aux ressources Amazon ECS à l'aide des balises de ressources" et Balisage de ressources .

Pour autoriser le balisage lors de la création, créez ou modifiez une politique afin d'inclure à la fois les autorisations d'utilisation de l'action qui crée la ressource, telle que `ecs:CreateCluster` ou `ecs:RunTask` et l'action `ecs:TagResource`.

L'exemple suivant illustre une politique qui permet aux utilisateurs de créer des clusters et d'ajouter des balises lors de la création du cluster. Les utilisateurs ne sont pas autorisés à attribuer des balises aux ressources existantes (ils ne peuvent pas appeler l'action `ecs:TagResource` directement).

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ecs:CreateCluster"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ecs:TagResource"
      ],
      "Resource": "*",
      "Condition": {
         "StringEquals": {
```

```
                    "ecs:CreateAction": [
                        "CreateCluster",
                        "CreateCapacityProvider",
                        "CreateService",
                        "CreateTaskSet",
                        "RegisterContainerInstance",
                        "RegisterTaskDefinition",
                        "RunTask",
                        "StartTask"
                    ]
            }
        }
    }
  ]
}
```

L'action `ecs:TagResource` est uniquement évaluée si les balises sont appliquées pendant l'action de création de ressources. Par conséquent, un utilisateur qui est autorisé à créer une ressource (en supposant qu'il n'existe aucune condition de balisage) n'a pas besoin des autorisations d'utiliser l'action `ecs:TagResource` si aucune balise n'est spécifié dans la demande. Toutefois, si l'utilisateur essaie de créer une ressource avec des balises, la demande échoue s'il n'a pas les autorisations d'utiliser l'action `ecs:TagResource`.

Amazon ECS contrôle l'accès à des balises spécifiques

Vous pouvez utiliser des conditions supplémentaires dans l'élément `Condition` de vos stratégies IAM pour contrôler les clés de balise et les valeurs qui peuvent être appliquées aux ressources.

Les clés de condition suivantes peuvent être utilisées avec les exemples de la section précédente :

- `aws:RequestTag` : Pour indiquer qu'une clé de balise ou une clé et valeur de balise particulière doit être présente dans une demande. D'autres balises peuvent également être spécifiées dans la demande.
  - Utilisez avec l'opérateur de condition `StringEquals` pour appliquer une combinaison de clé de balise et de valeur spécifique ; par exemple, pour appliquer la balise `cost-center=cc123` :

    ```
    "StringEquals": { "aws:RequestTag/cost-center": "cc123" }
    ```

  - A utiliser avec l'opération de condition `StringLike` pour appliquer une clé de balise spécifique dans la demande ; par exemple, pour appliquer la clé de balise `purpose`:

```
"StringLike": { "aws:RequestTag/purpose": "*" }
```

- `aws:TagKeys` : Pour appliquer les clés de balise qui sont utilisées dans la demande.
  - A utiliser avec le modificateur `ForAllValues` pour appliquer des clés de balise spécifiques si celles-ci sont fournies dans la demande (si les balises sont spécifiées dans la demande, seules les clés de balise spécifiques sont autorisées ; aucune autre balise n'est autorisée). Par exemple, les clés de balise `environment` ou `cost-center` sont autorisées :

    ```
    "ForAllValues:StringEquals": { "aws:TagKeys": ["environment","cost-center"] }
    ```

  - A utiliser avec le modificateur `ForAnyValue` pour appliquer la présence d'au moins l'une des clés de balise spécifiées dans la demande. Par exemple, au moins l'une des clés de balise `environment` ou `webserver` doit être présente dans la demande :

    ```
    "ForAnyValue:StringEquals": { "aws:TagKeys": ["environment","webserver"] }
    ```

Ces clés de condition peuvent être appliquées aux actions de création de ressources qui prennent en charge le balisage, ainsi qu'à l'action `ecs:TagResource`. Pour savoir si une action d'API Amazon ECS prend en charge le balisage, veuillez consulter Actions, ressources et clés de condition pour Amazon ECS.

Pour forcer les utilisateurs à spécifier des balises quand ils créent une ressource, vous devez utiliser la clé de condition `aws:RequestTag` ou la clé de condition `aws:TagKeys` avec le modificateur `ForAnyValue` sur l'action de création de ressources. L'action `ecs:TagResource` n'est pas évaluée si un utilisateur ne spécifie pas de balises pour l'action de création de ressources.

Pour les conditions, la clé de condition n'est pas sensible à la casse et la valeur de la condition est sensible à la casse. Par conséquent pour forcer la sensibilité à la casse d'une clé de balise, utilisez la clé de condition `aws:TagKeys`, où la clé de balise est indiquée comme une valeur dans la condition.

Pour plus d'informations sur les conditions à valeurs multiples, consultez la section Conditions comportant plusieurs clés ou valeurs de contexte dans le Guide de l'utilisateur IAM.

Contrôle de l'accès aux ressources Amazon ECS à l'aide des balises de ressources

Lorsque vous créez une politique IAM qui accorde aux utilisateurs l'autorisation d'utiliser les ressources Amazon ECS, vous pouvez inclure des informations de balise dans l'élément `Condition` de la politique pour contrôler l'accès en fonction des balises. Ceci est connu sous le nom de contrôle

d'accès basé sur les attributs (ABAC). ABAC vous offre un meilleur contrôle sur les ressources qu'un utilisateur peut modifier, utiliser ou supprimer. Pour plus d'informations, consultez Présentation d'ABAC pour AWS.

Par exemple, vous pouvez créer une politique qui permet aux utilisateurs de supprimer un cluster, mais qui refuse l'action si le cluster possède la balise `environment=production`. Pour ce faire, vous utilisez la clé de condition `aws:ResourceTag` pour autoriser ou refuser l'accès à la ressource en fonction des balises attachées à la ressource.

```
"StringEquals": { "aws:ResourceTag/environment": "production" }
```

Pour savoir si une action d'API Amazon ECS prend en charge le contrôle d'accès à l'aide de la clé de condition `aws:ResourceTag`, veuillez consulter Actions, ressources et clés de condition pour Amazon ECS. Notez que les actions `Describe` ne prennent pas en charge les autorisations au niveau des ressources, vous devez donc les spécifier dans une instruction distincte sans condition.

Par exemple les stratégies IAM, consultez Exemples de politiques Amazon ECS .

Si vous autorisez ou refusez à des utilisateurs l'accès à des ressources en fonction de balises, vous devez envisager de refuser de manière explicite la possibilité pour les utilisateurs d'ajouter ces balises ou de les supprimer des mêmes ressources. Sinon, il sera possible pour un utilisateur de contourner vos restrictions et d'obtenir l'accès à une ressource en modifiant ses balises.

Exemples de politiques Amazon ECS

Vous pouvez utiliser les politiques IAM pour octroyer aux utilisateurs les autorisations d'afficher et d'utiliser des ressources spécifiques sur la console Amazon ECS. Vous pouvez utiliser les exemples de politiques présentés dans la section précédente ; toutefois, ils sont conçus pour les demandes effectuées avec le AWS CLI ou un AWS SDK.

Exemple : autoriser les utilisateurs à supprimer un cluster Amazon ECS en fonction de balises

La politique suivante permet aux utilisateurs de supprimer des clusters lorsque la balise comporte une key/value paire « Objectif/Test ».

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Action": [
                "ecs:DeleteCluster"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ecs:us-east-1:111122223333:cluster/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Purpose": "Testing"
                }
            }
        }
    ]
}
```

Dépannage de l'identité et de l'accès Amazon Elastic Container Service

Utilisez les informations suivantes pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lorsque vous travaillez avec Amazon ECS et IAM.

Rubriques

- Je ne suis pas autorisé à effectuer une action dans Amazon ECS
- Je ne suis pas autorisé à effectuer iam : PassRole
- Je souhaite autoriser des personnes extérieures Compte AWS à moi à accéder à mes ressources Amazon ECS
- Ressources supplémentaires pour la résolution des problèmes

Je ne suis pas autorisé à effectuer une action dans Amazon ECS

Si vous recevez une erreur qui indique que vous n'êtes pas autorisé à effectuer une action, vos politiques doivent être mises à jour afin de vous permettre d'effectuer l'action.

L'exemple d'erreur suivant se produit quand l'utilisateur IAM `mateojackson` tente d'utiliser la console pour afficher des informations détaillées sur une ressource *`my-example-widget`* fictive, mais ne dispose pas des autorisations `ecs:`*`GetWidget`* fictives.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 ecs:GetWidget on resource: my-example-widget
```

Dans ce cas, la politique qui s'applique à l'utilisateur `mateojackson` doit être mise à jour pour autoriser l'accès à la ressource *`my-example-widget`* à l'aide de l'action `ecs:`*`GetWidget`*.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Je ne suis pas autorisé à effectuer iam : PassRole

Si vous recevez une erreur selon laquelle vous n'êtes pas autorisé à exécuter l'action `iam:PassRole`, vos politiques doivent être mises à jour pour vous permettre de transmettre un rôle à Amazon ECS.

Certains vous Services AWS permettent de transmettre un rôle existant à ce service au lieu de créer un nouveau rôle de service ou un rôle lié à un service. Pour ce faire, un utilisateur doit disposer des autorisations nécessaires pour transmettre le rôle au service.

L'exemple d'erreur suivant se produit lorsqu'un utilisateur IAM nommé `marymajor` essaie d'utiliser la console pour effectuer une action dans Amazon ECS. Toutefois, l'action nécessite que le service ait des autorisations accordées par un rôle de service. Mary ne dispose pas des autorisations nécessaires pour transférer le rôle au service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dans ce cas, les politiques de Mary doivent être mises à jour pour lui permettre d'exécuter l'action `iam:PassRole`.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Je souhaite autoriser des personnes extérieures Compte AWS à moi à accéder à mes ressources Amazon ECS

Vous pouvez créer un rôle que les utilisateurs provenant d'autres comptes ou les personnes extérieures à votre organisation pourront utiliser pour accéder à vos ressources. Vous pouvez spécifier qui est autorisé à assumer le rôle. Pour les services qui prennent en charge les politiques basées sur les ressources ou les listes de contrôle d'accès (ACLs), vous pouvez utiliser ces politiques pour autoriser les utilisateurs à accéder à vos ressources.

Pour plus d’informations, consultez les éléments suivants :

- Pour savoir si Amazon ECS est compatible avec ces fonctions, veuillez consulter Fonctionnement d'Amazon Elastic Container Service avec IAM.
- Pour savoir comment fournir l'accès à vos ressources sur celles Comptes AWS que vous possédez, consultez la section Fournir l'accès à un utilisateur IAM dans un autre utilisateur Compte AWS que vous possédez dans le Guide de l'utilisateur IAM.
- Pour savoir comment fournir l'accès à vos ressources à des tiers Comptes AWS, consultez la section Fournir un accès à des ressources Comptes AWS détenues par des tiers dans le guide de l'utilisateur IAM.
- Pour savoir comment fournir un accès par le biais de la fédération d’identité, consultez Fournir un accès à des utilisateurs authentifiés en externe (fédération d’identité) dans le Guide de l’utilisateur IAM.
- Pour en savoir plus sur la différence entre l’utilisation des rôles et des politiques basées sur les ressources pour l’accès intercompte, consultez Accès intercompte aux ressources dans IAM dans le Guide de l’utilisateur IAM.

Ressources supplémentaires pour la résolution des problèmes

Les pages suivantes fournissent des informations sur les codes d'erreur :

- Messages d'erreur relatifs aux tâches interrompues par Amazon ECS
- Affichage des messages relatifs aux événements du service Amazon ECS

Bonnes pratiques en matière d'IAM pour Amazon ECS

Vous pouvez utiliser AWS Identity and Access Management (IAM) pour gérer et contrôler l'accès à vos AWS services et ressources par le biais de politiques basées sur des règles à des fins d'authentification et d'autorisation. Plus précisément, grâce à ce service, vous contrôlez l'accès à vos AWS ressources à l'aide de politiques appliquées aux utilisateurs, aux groupes ou aux rôles. Parmi ces trois, les utilisateurs sont des comptes qui peuvent avoir accès à vos ressources. De plus, un rôle IAM est un ensemble d'autorisations qui peuvent être endossées par une identité authentifiée, qui n'est pas associée à une identité particulière en dehors d'IAM. Pour plus d'informations, consultez la présentation de la gestion des accès par Amazon ECS : autorisations et politiques.

Respect de la politique de moindre privilège d'accès

Créez des politiques dont le champ d'application est défini pour permettre aux utilisateurs d'effectuer les tâches qui leur sont prescrites. Par exemple, si un développeur doit régulièrement arrêter une tâche, créez une politique qui autorise uniquement cette action particulière. L'exemple suivant permet uniquement à un utilisateur d'arrêter une tâche qui appartient à une `task_family` spécifique dans un cluster avec un Amazon Resource Name (ARN) spécifique. La référence à un ARN dans une condition est également un exemple d'utilisation d'autorisations au niveau des ressources. Vous pouvez utiliser les autorisations au niveau des ressources pour spécifier la ressource à laquelle vous souhaitez qu'une action s'applique.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:StopTask"
            ],
            "Condition": {
                "ArnEquals": {
                       "ecs:cluster": "arn:aws:ecs:us-
east-1:111122223333:cluster/cluster_name"
                }
            },
            "Resource": [
                   "arn:aws:ecs:us-east-1:111122223333:task-
definition/task_family:*"
            ]
        }
    ]
}
```

Faire en sorte que les ressources du cluster servent de limite administrative

Les politiques dont le champ d'application est trop restreint peuvent entraîner une prolifération de rôles et augmenter les frais administratifs. Plutôt que de créer des rôles limités à des tâches ou à des

services spécifiques uniquement, créez des rôles délimités à des clusters et utilisez le cluster comme limite administrative principale.

Créez des pipelines automatisés pour isoler les utilisateurs finaux de l'API

Vous pouvez limiter les actions que les utilisateurs peuvent utiliser en créant des pipelines qui empaquettent et déploient automatiquement des applications sur des clusters Amazon ECS. Cela délègue efficacement la création, la mise à jour et la suppression de tâches au pipeline. Pour plus d'informations, consultez Tutoriel : déploiement standard d'Amazon ECS CodePipeline dans le guide de AWS CodePipeline l'utilisateur.

Utilisez les conditions de politiques pour renforcer la couche de sécurité

Lorsque vous avez besoin d'un niveau de sécurité supplémentaire, ajoutez une condition à votre politique. Cela peut être utile si vous effectuez une opération privilégiée ou lorsque vous devez restreindre l'ensemble des actions pouvant être effectuées sur des ressources spécifiques. L'exemple de politique suivant nécessite une autorisation multifactorielle lors de la suppression d'un cluster.

JSON

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:DeleteCluster"
      ],
      "Condition": {
        "Bool": {
          "aws:MultiFactorAuthPresent": "true"
        }
      },
    "Resource": ["*"]
    }
  ]
}
```

Les balises appliquées aux services sont propagées à toutes les tâches qui font partie de ce service. De ce fait, vous pouvez créer des rôles limités aux ressources Amazon ECS avec des balises

spécifiques. Dans la politique suivante, un directeur IAM démarre et arrête toutes les tâches avec une clé de balise `Department` et une valeur de balise de. `Accounting`

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
            "Resource": "arn:aws:ecs:arn:aws::111122223333:task/my-cluster/*",
            "Condition": {
                "StringEquals": {"ecs:ResourceTag/Department": "Accounting"}
            }
        }
    ]
}
```

Auditez régulièrement l'accès au APIs

Un utilisateur peut changer de rôle. Une fois son rôle changé, les autorisations qui lui ont été accordées précédemment risquent de ne plus s'appliquer. Assurez-vous de vérifier qui a accès à Amazon ECS APIs et de vérifier si cet accès est toujours justifié. Envisagez d'intégrer IAM à une solution de gestion du cycle de vie des utilisateurs qui révoque automatiquement l'accès lorsqu'un utilisateur quitte l'organisation. Pour plus d'informations, consultez les directives relatives aux audits de AWS sécurité dans le guide de AWS Identity and Access Management l'utilisateur.

Journalisation et surveillance dans Amazon Elastic Container Service

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances d'Amazon Elastic Container Service et de vos AWS solutions. Vous devez collecter des données de surveillance provenant de toutes les parties de votre AWS solution afin de pouvoir

corriger plus facilement une défaillance multipoint, le cas échéant. AWS fournit plusieurs outils pour surveiller vos ressources Amazon ECS et répondre aux incidents potentiels :

CloudWatch Alarmes Amazon

Surveillez une seule métrique sur une durée définie et exécutez une ou plusieurs actions en fonction de la valeur de la métrique par rapport à un seuil donné sur un certain nombre de durées. L'action est une notification envoyée à une rubrique Amazon Simple Notification Service (Amazon SNS) ou à une politique Amazon EC2 Auto Scaling. CloudWatch les alarmes n'appellent pas d'actions simplement parce qu'elles sont dans un état particulier ; l'état doit avoir changé et être maintenu pendant un certain nombre de périodes. Pour de plus amples informations, veuillez consulter Surveillez Amazon ECS à l'aide de CloudWatch.

Pour les services dont les tâches utilisent le type de lancement Fargate, vous pouvez CloudWatch utiliser des alarmes pour augmenter ou réduire les tâches de votre service en CloudWatch fonction de mesures telles que l'utilisation du processeur et de la mémoire. Pour de plus amples informations, veuillez consulter Faites évoluer automatiquement votre service Amazon ECS.

Pour les clusters dont les tâches ou les services utilisent le type de EC2 lancement, vous pouvez utiliser des CloudWatch alarmes pour augmenter ou réduire les instances de conteneur en fonction de CloudWatch métriques, telles que la réservation de mémoire du cluster.

Amazon CloudWatch Logs

Surveillez, stockez et accédez aux fichiers journaux à partir des conteneurs de vos tâches Amazon ECS en spécifiant le pilote de journal `awslogs` dans vos définitions de tâche. Pour plus d'informations, veuillez consulter Utilisation du pilote awslogs (langue française non garantie).

Vous pouvez également surveiller, stocker et accéder au système d'exploitation et aux fichiers journaux de l'agent de conteneur Amazon ECS à partir de vos instances de conteneur Amazon ECS. Cette méthode d'accès aux journaux peut être utilisée pour les conteneurs utilisant le type de EC2 lancement.

CloudWatch Événements Amazon

Mettez en correspondance les événements et transmettez-les à un ou plusieurs flux ou fonctions cibles pour effectuer des modifications, capturer des informations d'état et prendre des mesures correctives. Pour plus d'informations, consultez Automatisez les réponses aux erreurs Amazon ECS à l'aide de EventBridge ce guide et découvrez EventBridge l'évolution d'Amazon CloudWatch Events dans le guide de EventBridge l'utilisateur Amazon.

AWS CloudTrail Journaux

CloudTrail fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans Amazon ECS. À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande envoyée à Amazon ECS, l'adresse IP à partir de laquelle la demande a été faite, l'auteur de la demande, la date à laquelle elle a été faite, ainsi que des informations supplémentaires. Pour de plus amples informations, veuillez consulter Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail.

AWS Trusted Advisor

Trusted Advisor s'appuie sur les meilleures pratiques apprises en servant des centaines de milliers de AWS clients. Trusted Advisor inspecte votre AWS environnement, puis émet des recommandations lorsque des opportunités se présentent pour économiser de l'argent, améliorer la disponibilité et les performances du système ou contribuer à combler les failles de sécurité. Tous les AWS clients ont accès à cinq Trusted Advisor chèques. Les clients disposant d'un plan de support Business ou Enterprise peuvent consulter tous les Trusted Advisor chèques.

Pour plus d'informations, consultez AWS Trusted Advisor dans le Guide de l'utilisateur Support .

AWS Compute Optimizer

AWS Compute Optimizer est un service qui analyse les paramètres de configuration et d'utilisation de vos AWS ressources. Il indique si vos ressources sont optimales et génère des recommandations d'optimisation afin de réduire les coûts et d'améliorer les performances de vos charges de travail.

Pour de plus amples informations, veuillez consulter AWS Compute Optimizer recommandations pour Amazon ECS.

Un autre aspect important de la surveillance d'Amazon ECS consiste à surveiller manuellement les éléments non couverts par les CloudWatch alarmes. Le CloudWatch tableau de bord de AWS console et les autres tableaux de bord fournissent une at-a-glance vue de l'état de votre AWS environnement. Trusted Advisor Nous vous recommandons de vérifier également les fichiers journaux de vos instances de conteneur et les conteneurs de vos tâches.

Validation de la conformité pour Amazon Elastic Container Service

Pour savoir si un programme Services AWS de conformité Service AWS s'inscrit dans le champ d'application de programmes de conformité spécifiques, consultez Services AWS la section de

conformité et sélectionnez le programme de conformité qui vous intéresse. Pour des informations générales, voir Programmes de AWS conformité Programmes AWS de .

Vous pouvez télécharger des rapports d'audit tiers à l'aide de AWS Artifact. Pour plus d'informations, voir Téléchargement de rapports dans AWS Artifact .

Votre responsabilité en matière de conformité lors de l'utilisation Services AWS est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise et les lois et réglementations applicables. AWS fournit les ressources suivantes pour faciliter la mise en conformité :

- Conformité et gouvernance de la sécurité : ces guides de mise en œuvre de solutions traitent des considérations architecturales et fournissent les étapes à suivre afin de déployer des fonctionnalités de sécurité et de conformité.
- Référence des services éligibles HIPAA : liste les services éligibles HIPAA. Tous ne Services AWS sont pas éligibles à la loi HIPAA.
- AWS Ressources de https://aws.amazon.com/compliance/resources/ de conformité — Cette collection de classeurs et de guides peut s'appliquer à votre secteur d'activité et à votre région.
- AWS Guides de conformité destinés aux clients — Comprenez le modèle de responsabilité partagée sous l'angle de la conformité. Les guides résument les meilleures pratiques en matière de sécurisation Services AWS et décrivent les directives relatives aux contrôles de sécurité dans de nombreux cadres (notamment le National Institute of Standards and Technology (NIST), le Payment Card Industry Security Standards Council (PCI) et l'Organisation internationale de normalisation (ISO)).
- Évaluation des ressources à l'aide des règles du guide du AWS Config développeur : le AWS Config service évalue dans quelle mesure les configurations de vos ressources sont conformes aux pratiques internes, aux directives du secteur et aux réglementations.
- AWS Security Hub— Cela Service AWS fournit une vue complète de votre état de sécurité interne AWS. Security Hub utilise des contrôles de sécurité pour évaluer vos ressources AWS et vérifier votre conformité par rapport aux normes et aux bonnes pratiques du secteur de la sécurité. Pour obtenir la liste des services et des contrôles pris en charge, consultez Référence des contrôles Security Hub.
- Amazon GuardDuty — Cela Service AWS détecte les menaces potentielles qui pèsent sur vos charges de travail Comptes AWS, vos conteneurs et vos données en surveillant votre environnement pour détecter toute activité suspecte et malveillante. GuardDuty peut vous aider à répondre à diverses exigences de conformité, telles que la norme PCI DSS, en répondant aux exigences de détection des intrusions imposées par certains cadres de conformité.

- AWS Audit Manager— Cela vous Service AWS permet d'auditer en permanence votre AWS utilisation afin de simplifier la gestion des risques et la conformité aux réglementations et aux normes du secteur.

Bonnes pratiques en matière de conformité et de sécurité pour Amazon ECS

Lorsque vous utilisez Amazon ECS, votre responsabilité en matière de conformité dépend de la sensibilité de vos données, des objectifs de conformité de votre entreprise et des lois et réglementations applicables.

Normes de sécurité des données de l'industrie des cartes de paiement (PCI DSS)

Il est important que vous compreniez le flux complet des données des titulaires de cartes (CHD) au sein de l'environnement lorsque vous adhérez à la norme PCI DSS. Le flux CHD détermine l'applicabilité de la norme PCI DSS, définit les limites et les composants d'un environnement de données pour les titulaires de cartes (CDE) et, par conséquent, le champ d'application d'une évaluation PCI DSS. La détermination précise du champ d'application de la norme PCI DSS est essentielle pour définir la posture de sécurité et, en fin de compte, une évaluation réussie. Les clients doivent disposer d'une procédure de détermination du champ d'application garantissant son exhaustivité et détectant les modifications ou les écarts.

La nature temporaire des applications conteneurisées complique encore davantage l'audit des configurations. Par conséquent, les clients doivent rester informés de tous les paramètres de configuration des conteneurs afin de garantir le respect des exigences de conformité tout au long du cycle de vie d'un conteneur.

Pour plus d'informations sur la conformité à la norme PCI DSS sur Amazon ECS, consultez les livres blancs suivants.

- Architecture sur Amazon ECS pour la conformité à la norme PCI DSS

HIPAA (U.S. Health Insurance Portability and Accountability Act)

L'utilisation d'Amazon ECS avec des charges de travail qui traitent des informations de santé protégées (PHI) ne nécessite aucune configuration supplémentaire. Amazon ECS agit comme un service d'orchestration qui coordonne le lancement de conteneurs sur Amazon EC2. Il ne fonctionne

pas avec ou sur les données de la charge de travail orchestrée. Conformément aux réglementations HIPAA et à l' AWS Business Associate Addendum, les PHI doivent être chiffrées en transit et au repos lorsqu'elles sont consultées par des conteneurs lancés avec Amazon ECS.

Différents mécanismes de chiffrement au repos sont disponibles avec chaque option AWS de stockage, tels qu'Amazon S3, Amazon EBS et. AWS KMS Vous pouvez déployer un réseau superposé (tel que VNS3 Weave Net) pour garantir le chiffrement complet des données PHI transférées entre les conteneurs ou pour fournir une couche de chiffrement redondante. Vous devez également utiliser la journalisation complète et rediriger tous les journaux des conteneurs vers Amazon CloudWatch. Pour plus d'informations sur l'utilisation des meilleures pratiques en matière de sécurité de l'infrastructure, voir Protection de l'infrastructure dans le cadre AWS bien architecturé du pilier de sécurité.

AWS Security Hub

Utiliser AWS Security Hub. Cela Service AWS fournit une vue complète de votre état de sécurité interne AWS. Security Hub utilise des contrôles de sécurité pour évaluer vos ressources AWS et vérifier votre conformité par rapport aux normes et aux bonnes pratiques du secteur de la sécurité. Pour obtenir la liste des services et des contrôles pris en charge, consultez Référence des contrôles Security Hub.

Amazon GuardDuty avec Amazon ECS Runtime Monitoring

Amazon GuardDuty est un service de détection des menaces qui aide à protéger vos comptes, vos conteneurs, vos charges de travail et les données de votre AWS environnement. À l'aide de modèles d'apprentissage automatique (ML) et de capacités de détection des anomalies et des menaces, vous surveillez GuardDuty en permanence les différentes sources de journaux et l'activité d'exécution afin d'identifier et de hiérarchiser les risques de sécurité potentiels et les activités malveillantes dans votre environnement.

Utilisez Runtime Monitoring GuardDuty pour identifier les comportements malveillants ou non autorisés. La surveillance du temps d'exécution protège les charges de travail exécutées sur Fargate EC2 et en AWS surveillant en permanence l'activité des journaux et du réseau afin d'identifier les comportements malveillants ou non autorisés. Runtime Monitoring utilise un agent de GuardDuty sécurité léger et entièrement géré qui analyse le comportement sur l'hôte, tel que l'accès aux fichiers, l'exécution des processus et les connexions réseau. Cela couvre des problèmes tels que l'augmentation des privilèges, l'utilisation d'informations d'identification divulguées ou la communication avec des adresses IP ou des domaines malveillants, ainsi que la présence de

logiciels malveillants sur vos EC2 instances Amazon et vos charges de travail de conteneur. Pour plus d'informations, consultez la section Surveillance du temps GuardDuty d'exécution dans le guide de GuardDuty l'utilisateur.

Recommandations en matière de conformité

Nous vous recommandons d'impliquer rapidement les responsables des programmes de conformité au sein de votre entreprise et d'utiliser le modèle de responsabilité AWS partagée pour identifier les responsables du contrôle de conformité afin de garantir le succès des programmes de conformité pertinents. Pour de plus amples informations, veuillez consulter AWS modèle de responsabilité partagée pour Amazon ECS .

AWS Fargate Norme fédérale de traitement de l'information (FIPS-140)

Norme Federal Information Processing Standard (FIPS). La norme FIPS-140 est une norme des gouvernements américain et canadien qui spécifie les exigences de sécurité pour les modules cryptographiques qui protègent des informations sensibles. La norme FIPS-140 définit un ensemble de fonctions cryptographiques validées qui peuvent être utilisées pour chiffrer les données en transit et les données au repos.

Lorsque vous activez la conformité à la norme FIPS-140, vous pouvez exécuter des charges de travail sur Fargate conformément à cette norme. Pour plus d'informations sur la conformité à la norme FIPS-140, consultez la norme fédérale de traitement de l'information (FIPS) 140-3.

AWS Fargate Considérations relatives à la norme FIPS-140

Prenez en compte les éléments suivants lors de l'utilisation de la norme FIPS-140 sur Fargate :

- La conformité à la norme FIPS-140 n'est disponible que dans les régions AWS GovCloud (US) .
- Fargate prend en charge la version 140.3 de la norme FIPS-140
- Elle est désactivée par défaut. Vous devez l'activer.
- Vos tâches doivent utiliser la configuration suivante pour garantir la conformité à la norme FIPS-140 :
  - Le `operatingSystemFamily` doit être `LINUX`.
  - Le `cpuArchitecture` doit être `X86_64`.

- La version de plateforme Fargate doit être `1.4.0` ou ultérieure.

Utilisation de FIPS sur Fargate

Suivez cette procédure pour utiliser FIPS-140 sur Fargate.

1. Activez la conformité à la norme FIPS-140. Pour de plus amples informations, veuillez consulter the section called “AWS Fargate Conformité à la norme fédérale de traitement de l'information (FIPS-140)”.
2. Vous pouvez éventuellement utiliser ECS Exec pour exécuter la commande suivante afin de vérifier l'état de conformité à la norme FIPS-140 d'un cluster.

   Remplacez-le *`cluster-name`* par le nom de votre cluster, *`task-id`* par l'ID ou l'ARN de votre tâche et *`container-name`* par le nom du conteneur de votre tâche sur lequel vous souhaitez exécuter la commande.

   Une valeur renvoyée de « 1 » indique que vous utilisez FIPS.

   ```
   aws ecs execute-command \
       --cluster cluster-name \
       --task task-id \
       --container container-name \
       --interactive \
       --command "cat /proc/sys/crypto/fips_enabled"
   ```

Utilisation CloudTrail pour l'audit Fargate FIPS-140

CloudTrail est activé dans votre AWS compte lorsque vous le créez. Lorsque l'activité de l'API et de la console se produit dans Amazon ECS, cette activité est enregistrée dans un CloudTrail événement avec d'autres événements de AWS service dans l'historique des événements. Vous pouvez consulter, rechercher et télécharger les événements récents dans votre AWS compte. Pour plus d'informations, consultez la section Affichage des événements avec l'historique des CloudTrail événements.

Pour un enregistrement continu des événements de votre AWS compte, y compris des événements relatifs à Amazon ECS, créez un journal qui CloudTrail sera utilisé pour envoyer les fichiers journaux dans un compartiment Amazon S3. Par défaut, lorsque vous créez un journal d’activité dans la console, il s’applique à toutes les régions. Le journal enregistre les événements de toutes les régions

de la AWS partition et transmet les fichiers journaux au compartiment Amazon S3 que vous spécifiez. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en détail les données d'événements collectées dans les CloudTrail journaux et agir en conséquence. Pour de plus amples informations, veuillez consulter the section called “Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail”.

L'exemple suivant montre une entrée de CloudTrail journal qui illustre l'action de l'`PutAccountSettingDefault`API :

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAIV5AJI5LXF5EXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/jdoe",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIPWIOFC3EXAMPLE",
    },
    "eventTime": "2023-03-01T21:45:18Z",
    "eventSource": "ecs.amazonaws.com",
    "eventName": "PutAccountSettingDefault",
    "awsRegion": "us-gov-east-1",
    "sourceIPAddress": "52.94.133.131",
    "userAgent": "aws-cli/2.9.8 Python/3.9.11 Windows/10 exe/AMD64 prompt/off command/
ecs.put-account-setting",
    "requestParameters": {
        "name": "fargateFIPSMode",
        "value": "enabled"
    },
    "responseElements": {
        "setting": {
            "name": "fargateFIPSMode",
            "value": "enabled",
            "principalArn": "arn:aws:iam::123456789012:user/jdoe"
        }
    },
    "requestID": "acdc731e-e506-447c-965d-f5f75EXAMPLE",
    "eventID": "6afced68-75cd-4d44-8076-0beEXAMPLE",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
```

```
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "ecs-fips.us-gov-east-1.amazonaws.com"
    }
}
```

Sécurité de l'infrastructure dans Amazon Elastic Container Service

En tant que service géré, Amazon Elastic Container Service est protégé par la sécurité du réseau AWS mondial. Pour plus d'informations sur les services AWS de sécurité et sur la manière dont AWS l'infrastructure est protégée, consultez la section Sécurité du AWS cloud. Pour concevoir votre AWS environnement en utilisant les meilleures pratiques en matière de sécurité de l'infrastructure, consultez la section Protection de l'infrastructure dans le cadre AWS bien architecturé du pilier de sécurité.

Vous utilisez des appels d'API AWS publiés pour accéder à Amazon ECS via le réseau. Les clients doivent prendre en charge les éléments suivants :

- Protocole TLS (Transport Layer Security). Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Ses suites de chiffrement PFS (Perfect Forward Secrecy) comme DHE (Ephemeral Diffie-Hellman) ou ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l’aide d’un ID de clé d’accès et d’une clé d’accès secrète associée à un principal IAM. Vous pouvez également utiliser AWS Security Token Service (AWS STS) pour générer des informations d’identification de sécurité temporaires et signer les demandes.

Vous pouvez appeler ces opérations d'API à partir de n'importe quel endroit du réseau. Amazon ECS prend en charge les stratégies d'accès basées sur les ressources, ce qui peut inclure des restrictions en fonction de l'adresse IP source. Assurez-vous donc que les politiques tiennent compte de l'adresse IP de l'emplacement du réseau. Vous pouvez également utiliser les politiques Amazon ECS pour contrôler l'accès depuis des points de terminaison Amazon Virtual Private Cloud spécifiques ou spécifiques VPCs. En fait, cela isole l'accès réseau à une ressource Amazon ECS donnée uniquement du VPC spécifique au sein AWS du réseau. Pour de plus amples informations, veuillez consulter Points de terminaison d'un VPC d'interface Amazon ECS (AWS PrivateLink).

Points de terminaison d'un VPC d'interface Amazon ECS (AWS PrivateLink)

Vous pouvez améliorer le niveau de sécurité de votre VPC en configurant Amazon ECS pour utiliser un point de terminaison d'un VPC d'interface. Les points de terminaison de l'interface sont AWS PrivateLink alimentés par une technologie qui vous permet d'accéder de manière privée à Amazon ECS à APIs l'aide d'adresses IP privées. AWS PrivateLink restreint tout le trafic réseau entre votre VPC et Amazon ECS vers le réseau Amazon. Vous n'avez pas besoin d'une passerelle Internet, d'un périphérique NAT ni d'une passerelle privée virtuelle.

Pour plus d'informations sur AWS PrivateLink les points de terminaison VPC, consultez la section Points de terminaison VPC dans le guide de l'utilisateur Amazon VPC.

Considérations

Considérations relatives aux terminaux dans les régions introduites à compter du 23 décembre 2023

Avant de configurer des points de terminaison d'un VPC d'interface pour Amazon ECS, tenez compte de ce qui suit :

- Vous devez disposer des points de terminaison VPC spécifiques à la région suivants :

  > **Note**
  >
  > Si vous ne configurez pas tous les points de terminaison, votre trafic passera par les points de terminaison publics, et non par votre point de terminaison VPC.

  - `com.amazonaws.region.ecs-agent`
  - `com.amazonaws.region.ecs-telemetry`
  - `com.amazonaws.region.ecs`

  Par exemple, la région du Canada Ouest (Calgary) (ca-west-1) a besoin des points de terminaison VPC suivants :

  - `com.amazonaws.ca-west-1.ecs-agent`
  - `com.amazonaws.ca-west-1.ecs-telemetry`
  - `com.amazonaws.ca-west-1.ecs`

- Lorsque vous utilisez un modèle pour créer AWS des ressources dans la nouvelle région et que le modèle a été copié à partir d'une région créée avant le 23 décembre 2023, en fonction de la région de copie, effectuez l'une des opérations suivantes.

  Par exemple, la région de copie est USA Est (Virginie du Nord) (us-east-1). La région de copie est Canada-Ouest (Calgary) (ca-west-1).

Configuration	Action
La région copiée ne possède aucun point de terminaison VPC.	Créez les trois points de terminaison VPC pour la nouvelle région (par exemple,). `com.amazonaws.ca-west-1.ecs-agent`
La région copiée contient des points de terminaison VPC spécifiques à la région.	a. Créez les trois points de terminaison VPC pour la nouvelle région (par exemple,). `com.amazonaws.ca-west-1.ecs-agent` b. Supprimez les trois points de terminaison VPC de la région de copie (par exemple,). `com.amazonaws.us-east-1.ecs-agent`

Éléments à prendre en compte pour les points de terminaison d'un VPC Amazon ECS pour le type de lancement Fargate

Lorsqu'il existe un point de terminaison VPC pour `ecr.dkr` et `ecr.api` dans le même VPC dans lequel une tâche Fargate est déployée, celui-ci utilise le point de terminaison VPC. S'il n'y a pas de point de terminaison VPC, celui-ci utilisera l'interface Fargate.

Avant de configurer des points de terminaison d'un VPC d'interface pour Amazon ECS, tenez compte de ce qui suit :

- Les tâches utilisant le type de lancement Fargate ne nécessitent pas les points de terminaison VPC d'interface pour Amazon ECS, mais vous pourriez avoir besoin de points de terminaison VPC d'interface pour Amazon ECR, Secrets Manager ou Amazon Logs décrits dans les points suivants. CloudWatch
  - Pour autoriser vos tâches à extraire des images privées depuis Amazon ECR, vous devez créer les points de terminaison d'un VPC d'interface pour Amazon ECR. Pour de plus amples informations, veuillez consulter Points de terminaison de VPC d'interface (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.

    > **Important**
    >
    > Si vous configurez Amazon ECR pour utiliser un point de terminaison de VPC d'interface, vous pouvez créer un rôle d'exécution de tâche incluant des clés de condition pour restreindre l'accès à un VPC ou à un point de terminaison de VPC spécifique. Pour de plus amples informations, veuillez consulter Tâches Fargate extrayant des images Amazon ECR au-delà des autorisations des points de terminaison de l'interface.

  - Pour autoriser vos tâches à extraire des données sensibles depuis Secrets Manager, vous devez créer les points de terminaison d'un VPC d'interface pour Secrets Manager. Pour de plus amples informations, veuillez consulter Utilisation de Secrets Manager avec des points de terminaison de VPC dans le Guide de l'utilisateur AWS Secrets Manager .
  - Si votre VPC ne possède pas de passerelle Internet et que vos tâches utilisent le pilote de journal pour envoyer des informations de `awslogs` journal aux CloudWatch journaux, vous devez créer un point de terminaison VPC d'interface pour les journaux. CloudWatch Pour plus d'informations, consultez la section Utilisation CloudWatch des journaux avec les points de terminaison VPC d'interface dans le guide de l'utilisateur Amazon CloudWatch Logs.
- Les points de terminaison d'un VPC ne prennent pas en charge les demandes inter-régions pour le moment. Veillez à créer votre point de terminaison dans la même région que celle dans laquelle vous souhaitez envoyer vos appels d'API à Amazon ECS. Supposons, par exemple, que vous souhaitiez exécuter des tâches dans USA Est (Virginie du Nord). Vous devez ensuite créer le point de terminaison Amazon ECS VPC dans USA Est (Virginie du Nord). Un point de terminaison d'un VPC Amazon ECS créé dans une autre région ne peut pas exécuter de tâche dans USA Est (Virginie du Nord).

- Les points de terminaison d'un VPC prennent uniquement en charge le DNS fourni par Amazon via Amazon Route 53. Si vous souhaitez utiliser votre propre DNS, vous pouvez utiliser le transfert DNS conditionnel. Pour en savoir plus, consultez Jeux d'options DHCP dans le Guide de l'utilisateur Amazon VPC.
- Le groupe de sécurité attaché au point de terminaison d'un VPC doit autoriser les connexions entrantes sur le port TCP 443 à partir du sous-réseau privé du VPC.
- La gestion Service Connect du proxy Envoy utilise le point de terminaison d'un VPC `com.amazonaws.region.ecs-agent`. Lorsque vous n'utilisez pas les points de terminaison d'un VPC, la gestion Service Connect du proxy Envoy utilise le point de terminaison `ecs-sc` de cette région. Pour obtenir la liste des points de terminaison Amazon ECS dans chaque région, veuillez consulter Points de terminaison et quotas Amazon ECS (langue française non garantie).

Considérations relatives aux points de terminaison VPC Amazon ECS pour le type de lancement EC2

Avant de configurer des points de terminaison d'un VPC d'interface pour Amazon ECS, tenez compte de ce qui suit :

- Les tâches utilisant le type de EC2 lancement nécessitent que les instances de conteneur sur lesquelles elles sont lancées exécutent la version `1.25.1` ou une version ultérieure de l'agent de conteneur Amazon ECS. Pour de plus amples informations, veuillez consulter Gestion des instances de conteneurs Linux Amazon ECS.
- Pour autoriser vos tâches à extraire des données sensibles depuis Secrets Manager, vous devez créer les points de terminaison d'un VPC d'interface pour Secrets Manager. Pour de plus amples informations, veuillez consulter Utilisation de Secrets Manager avec des points de terminaison de VPC dans le Guide de l'utilisateur AWS Secrets Manager .
- Si votre VPC ne possède pas de passerelle Internet et que vos tâches utilisent le pilote de journal pour envoyer des informations de `awslogs` journal aux CloudWatch journaux, vous devez créer un point de terminaison VPC d'interface pour les journaux. CloudWatch Pour plus d'informations, consultez la section Utilisation CloudWatch des journaux avec les points de terminaison VPC d'interface dans le guide de l'utilisateur Amazon CloudWatch Logs.
- Les points de terminaison d'un VPC ne prennent pas en charge les demandes inter-régions pour le moment. Veillez à créer votre point de terminaison dans la même région que celle dans laquelle vous souhaitez envoyer vos appels d'API à Amazon ECS. Supposons, par exemple, que vous souhaitiez exécuter des tâches dans USA Est (Virginie du Nord). Vous devez ensuite créer le point de terminaison Amazon ECS VPC dans USA Est (Virginie du Nord). Un point de terminaison VPC

Amazon ECS créé dans une autre région ne peut pas exécuter de tâches dans l'est des États-Unis (Virginie du Nord).

- Les points de terminaison d'un VPC prennent uniquement en charge le DNS fourni par Amazon via Amazon Route 53. Si vous souhaitez utiliser votre propre DNS, vous pouvez utiliser le transfert DNS conditionnel. Pour en savoir plus, consultez Jeux d'options DHCP dans le Guide de l'utilisateur Amazon VPC.
- Le groupe de sécurité attaché au point de terminaison d'un VPC doit autoriser les connexions entrantes sur le port TCP 443 à partir du sous-réseau privé du VPC.

Comprendre les modèles de dénomination des points de terminaison Amazon ECS

Il est important de comprendre que l'agent Amazon ECS peut envoyer des demandes à des points de terminaison dotés de suffixes numérotés, tels que :

- `ecs-a-1.region.amazonaws.comecs-a-2.region.amazonaws.com`, etc. pour les points de terminaison des agents
- `ecs-t-1.region.amazonaws.comecs-t-2.region.amazonaws.com`, etc. pour les points de terminaison de télémétrie

Ce comportement se produit parce que l'agent Amazon ECS utilise l'DiscoverPollEndpointAPI pour déterminer dynamiquement à quel point de terminaison spécifique se connecter. Si les points de terminaison de votre VPC ne gèrent pas correctement ces variations de points de terminaison numérotés, l'agent utilisera à nouveau des points de terminaison publics, même si vous avez configuré des points de terminaison VPC pour les noms de base.

Le rôle de l' DiscoverPollEndpointAPI

L'DiscoverPollEndpointAPI est utilisée par l'agent Amazon ECS pour découvrir le point de terminaison approprié pour interroger les tâches. Lorsque l'agent appelle cette API, il reçoit une URL de point de terminaison spécifique qui peut inclure un suffixe numéroté. Pour garantir le bon fonctionnement de vos points de terminaison VPC, la configuration de votre réseau doit permettre à l'agent de :

1. Accédez à l' DiscoverPollEndpoint API
2. Connectez-vous au point de terminaison renvoyé URLs, y compris à ceux dont les suffixes sont numérotés

Si vous résolvez des problèmes de connectivité des points de terminaison VPC, vérifiez que votre agent peut atteindre à la fois les points de terminaison de base et toutes les variantes numérotées susceptibles d'être renvoyées par l'API. DiscoverPollEndpoint

Création de points de terminaison de VPC pour Amazon ECS

Pour créer le point de terminaison VPC pour le service Amazon ECS, utilisez la procédure Access an AWS service using an interface VPC endpoint du guide de l'utilisateur Amazon VPC pour créer les points de terminaison suivants. Si vous disposez d'instances de conteneur dans votre VPC, vous devriez créer les points de terminaison dans l'ordre dans lequel ils sont répertoriés. Si vous envisagez de créer vos instances de conteneur après la création de votre point de terminaison d'un VPC, alors l'ordre n'est pas important.

> **Note**
>
> Si vous ne configurez pas tous les points de terminaison, votre trafic passera par les points de terminaison publics, et non par votre point de terminaison VPC.
> Lorsque vous créez des points de terminaison, Amazon ECS crée également un nom DNS privé pour le point de terminaison. Par exemple, pour ecs-agent et `ecs-a.region.amazonaws.com` `ecs-t.region.amazonaws.com` pour ecs-telemetry.

- `com.amazonaws.`*`region`*`.ecs-agent`
- `com.amazonaws.`*`region`*`.ecs-telemetry`
- `com.amazonaws.`*`region`*`.ecs`

> **Note**
>
> *`region`*représente l'identifiant de région d'une AWS région prise en charge par Amazon ECS, par exemple `us-east-2` pour la région USA Est (Ohio).

Le `ecs-agent` point de terminaison utilise l'`ecs:poll`API, et le `ecs-telemetry` point de terminaison utilise l'`ecs:StartTelemetrySession`API `ecs:poll` and.

Si vous avez des tâches existantes qui utilisent le type de EC2 lancement, une fois que vous avez créé les points de terminaison VPC, chaque instance de conteneur doit récupérer la nouvelle configuration. Pour ce faire, vous devez redémarrer chaque instance de conteneur ou redémarrer

l'agent de conteneur Amazon ECS sur chaque instance de conteneur. Faites ce qui suit pour redémarrer l'agent de conteneur.

Pour redémarrer l'agent de conteneur Amazon ECS

1. Connectez-vous à votre instance de conteneur via SSH.
2. Arrêtez l'agent de conteneur .

```
sudo docker stop ecs-agent
```

3. Démarrer l'agent de conteneur.

```
sudo docker start ecs-agent
```

Une fois les points de terminaison d'un VPC créés et l'agent de conteneur Amazon ECS redémarré sur chaque instance de conteneur, toutes les tâches nouvellement lancées adopteront la nouvelle configuration.

Création d'une stratégie de point de terminaison d'un VPC pour Amazon ECS

Vous pouvez attacher une stratégie de point de terminaison au point de terminaison d'un VPC qui contrôle l'accès à Amazon ECS. La politique spécifie les informations suivantes :

- Le principal qui peut exécuter des actions.
- Les actions qui peuvent être effectuées.
- Les ressources sur lesquelles les actions peuvent être exécutées.

Pour plus d'informations, consultez Contrôle de l'accès aux services avec points de terminaison d'un VPC dans le Guide de l'utilisateur Amazon VPC.

Exemple : stratégie de point de terminaison d'un VPC pour les actions de l'API Amazon ECS

Voici un exemple de politique de point de terminaison pour Amazon ECS. Lorsqu'elle est attachée à un point de terminaison, cette politique accorde l'accès à l'autorisation de créer et de répertorier des clusters. Les actions `CreateCluster` et `ListClusters` n'acceptent aucune ressource. La définition de ressource est donc définie sur * pour toutes les ressources.

```
{
   "Statement":[
```

```
    {
      "Principal":"*",
      "Effect": "Allow",
      "Action": [
        "ecs:CreateCluster",
        "ecs:ListClusters"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

AWS modèle de responsabilité partagée pour Amazon ECS

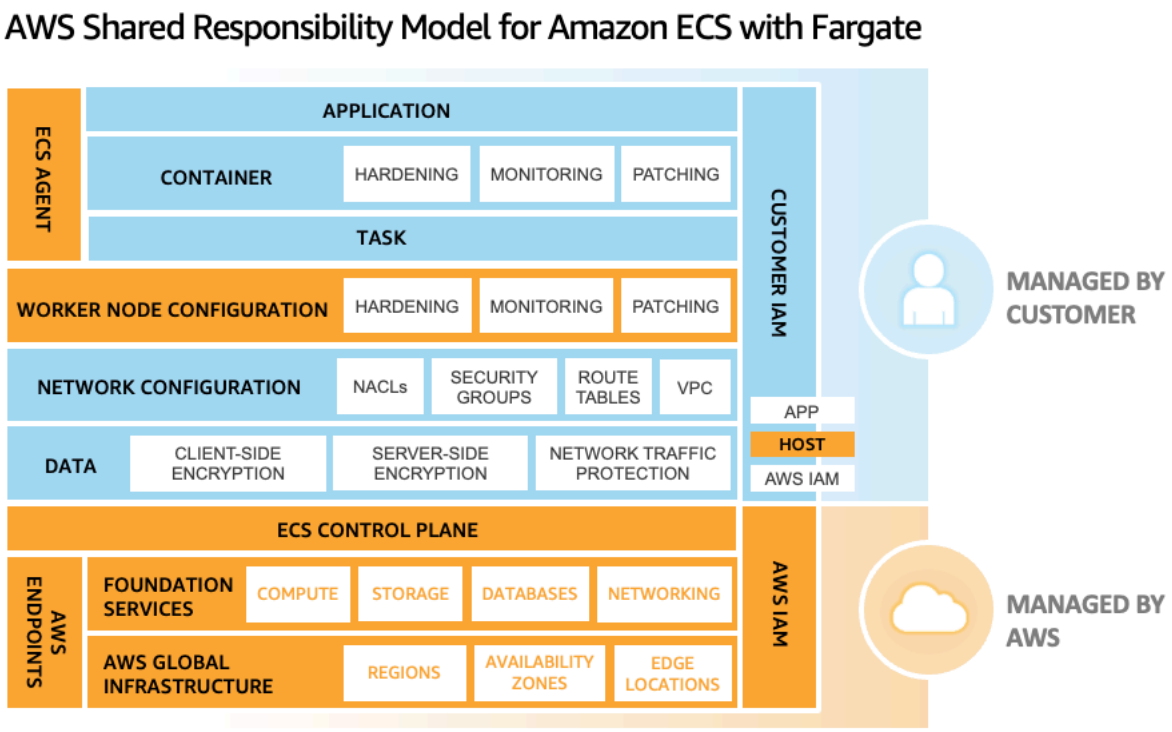
La sécurité et la conformité sont une responsabilité partagée entre le client AWS et le client. Ce modèle partagé peut contribuer à alléger la charge opérationnelle du client en AWS exploitant, en gérant et en contrôlant les composants, depuis le système d'exploitation hôte et la couche de virtualisation jusqu'à la sécurité physique des installations dans lesquelles le service fonctionne. Le client assume la responsabilité et la gestion du système d'exploitation client (y compris les mises à jour et les correctifs de sécurité), des autres logiciels d'application associés ainsi que de la configuration du pare-feu du groupe de sécurité AWS fourni. Les clients doivent choisir avec soin les services, car leurs responsabilités varient en fonction des services utilisés, de l'intégration de ces services dans leur environnement informatique, ainsi que les cadres législatifs et règlementaires applicables. La nature de cette responsabilité partagée assure également la flexibilité et le contrôle client qui permettent le déploiement.

Type de lancement Fargate

L'illustration suivante montre le modèle de responsabilité partagée pour le type de lancement Fargate. Fargate exécute chaque charge de travail dans un environnement matériel virtualisé isolé. Par conséquent, chaque tâche bénéficie d'une capacité d'infrastructure dédiée. Les charges de travail conteneurisées exécutées sur Fargate ne partagent pas de système d'exploitation, de noyau Linux, d'interface réseau, de stockage éphémère, de processeur ou de mémoire avec d'autres tâches. Lorsqu'ils utilisent Fargate, les clients ne sont pas responsables de la sécurisation de l'infrastructure informatique qui gère leurs conteneurs. Fargate fournira et corrigera l'infrastructure sur laquelle s'exécutent les charges de travail des clients. Pour de plus amples informations, veuillez consulter Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS .

Vous êtes responsable de la gestion des ressources suivantes :

- Configuration réseau, y compris le VPC NACLs, les groupes de sécurité et les tables de routage
- Chiffrement du stockage des clients et des services. Pour de plus amples informations, veuillez consulter Options de stockage pour les tâches Amazon ECS.
- Images du conteneur. Pour de plus amples informations, veuillez consulter Bonnes pratiques en matière de sécurité des tâches et des conteneurs Amazon ECS.
- Autorisations IAM pour les applications à l'aide du rôle de tâche. Pour de plus amples informations, veuillez consulter Rôle IAM de la tâche Amazon ECS.

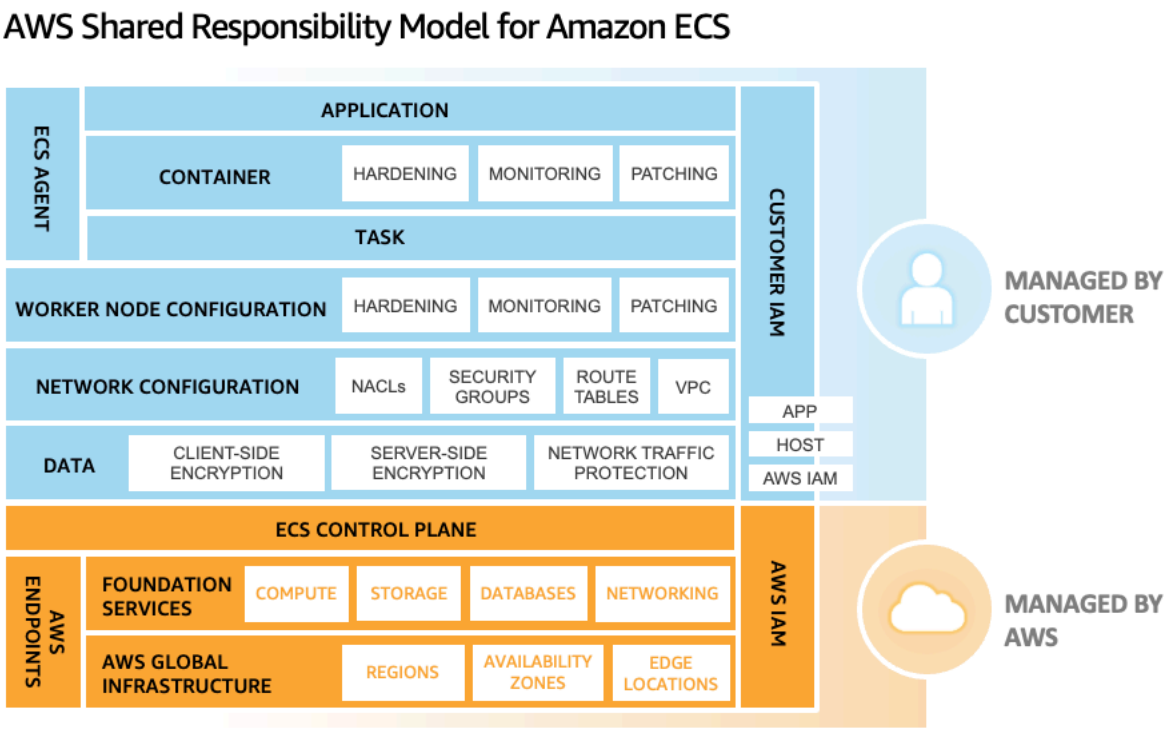


EC2 type de lancement

L'illustration suivante montre la responsabilité partagée pour le type de EC2 lancement. Lorsque vous exécutez des tâches sur EC2 des instances, vous êtes responsable de la maintenance de celles-ci en plus des ressources suivantes : EC2

- L'agent Amazon ECS.
- • L'AMI de l' EC2 instance, y compris les correctifs et le renforcement.
- Configuration réseau, y compris le VPC NACLs, les groupes de sécurité et les tables de routage.

- Chiffrement du stockage des clients et des services. Pour de plus amples informations, veuillez consulter Options de stockage pour les tâches Amazon ECS.
- Images du conteneur. Pour de plus amples informations, veuillez consulter Bonnes pratiques en matière de sécurité des tâches et des conteneurs Amazon ECS.
- Autorisations IAM pour les applications à l'aide du rôle de tâche. Pour de plus amples informations, veuillez consulter Rôle IAM de la tâche Amazon ECS.



Bonnes pratiques en matière de sécurité réseau pour Amazon ECS

La sécurité des réseaux est un vaste sujet qui englobe plusieurs sous-thèmes. Il s'agit encryption-in-transit notamment de la segmentation et de l'isolation du réseau, du pare-feu, du routage du trafic et de l'observabilité.

Chiffrement en transit

Le chiffrement du trafic réseau empêche les utilisateurs non autorisés d'intercepter et de lire des données lorsque celles-ci sont transmises sur un réseau. Avec Amazon ECS, le chiffrement réseau peut être mis en œuvre de l'une des façons suivantes.

- À l'aide des instances Nitro :

Par défaut, le trafic est automatiquement chiffré entre les types d'instances Nitro suivants : C5n, G4, I3en, M5dn, M5n, P3dn, R5dn et R5n. Le trafic n'est pas chiffré lorsqu'il est acheminé via une passerelle de transit, un équilibreur de charge ou un intermédiaire similaire.

- Chiffrement en transit
- Annonce des nouveautés de 2019
- Discussion lors de re:Inforce 2019

- À l'aide de Server Name Indication (SNI) et d'un Application Load Balancer :

  L'Application Load Balancer (ALB) et le Network Load Balancer (NLB) prennent en charge Server Name Indication (SNI). En utilisant SNI, vous pouvez placer plusieurs applications sécurisées derrière un seul écouteur. Pour cela, chaque écouteur possède son propre certificat TLS. Nous vous recommandons de fournir des certificats pour l'équilibreur de charge à l'aide d' AWS Certificate Manager (ACM), puis de les ajouter à la liste des certificats de l'écouteur. L'équilibreur de AWS charge utilise un algorithme de sélection de certificats intelligent avec SNI. Si le nom d'hôte fourni par un client correspond à un seul certificat de la liste de certificats, l'équilibreur de charge sélectionne ce certificat. Si un nom d'hôte fourni par un client correspond à plusieurs certificats de la liste, l'équilibreur de charge sélectionne un certificat pouvant être pris en charge par le client. Les exemples incluent un certificat auto-signé ou un certificat généré via l'ACM.

  - SNI avec un Application Load Balancer
  - SNI avec un Network Load Balancer

- End-to-end chiffrement à l'aide de certificats TLS :

  Cela implique le déploiement d'un certificat TLS avec la tâche. Il peut s'agir d'un certificat auto-signé ou d'un certificat délivré par une autorité de certification fiable. Vous pouvez obtenir le certificat en indiquant un secret pour le certificat. Sinon, vous pouvez choisir d'exécuter un conteneur qui envoie une demande de signature de certificat (CSR) à ACM, puis monte le secret obtenu sur un volume partagé.

  - Maintaining transport layer security all the way to your containers using the Network Load Balancer with Amazon ECS part 1
  - Maintien de la sécurité de la couche de transport (TLS) jusqu'à votre conteneur, partie 2 : Utilisation AWS Private Certificate Authority

Mise en réseau des tâches

Les recommandations suivantes tiennent compte du fonctionnement d'Amazon ECS. Amazon ECS n'utilise pas de réseau superposé. Au lieu de cela, les tâches sont configurées pour fonctionner dans différents modes réseau. Par exemple, les tâches configurées pour utiliser le mode `bridge` acquièrent une adresse IP non routable auprès d'un réseau Docker exécuté sur chaque hôte. Les tâches configurées pour utiliser le mode réseau `awsvpc` acquièrent une adresse IP auprès du sous-réseau de l'hôte. Les tâches configurées avec la mise en réseau `host` utilisent l'interface réseau de l'hôte. `awsvpc` est le mode réseau préféré. C'est parce qu'il s'agit du seul mode que vous pouvez utiliser pour attribuer des groupes de sécurité aux tâches. C'est également le seul mode disponible pour les AWS Fargate tâches sur Amazon ECS.

Groupes de sécurité pour les tâches

Nous vous recommandons de configurer vos tâches de manière à utiliser le mode réseau `awsvpc`. Une fois que vous avez configuré votre tâche pour utiliser ce mode, l'agent Amazon ECS provisionne et attache automatiquement une Interface réseau Elastic (ENI) à la tâche. Lorsque l'ENI est provisionné, la tâche est inscrite dans un groupe AWS de sécurité. Le groupe de sécurité agit en tant que pare-feu virtuel que vous pouvez utiliser afin de contrôler le trafic entrant et sortant.

Si vous utilisez un pare-feu personnalisé avec des tâches ou des services, ajoutez une règle sortante pour autoriser le trafic vers les points de terminaison de gestion des agents Amazon ECS (» `ecs-a-*.`*`region`*`.amazonaws.com` «), les points de terminaison de télémétrie (» `ecs-t-*.`*`region`*`.amazonaws.com` «) et les points de terminaison de gestion Service Connect Envoy (» «). `ecs-sc.`*`region`*`.api.aws`

AWS PrivateLink et Amazon ECS

AWS PrivateLink est une technologie réseau qui vous permet de créer des points de terminaison privés pour différents AWS services, notamment Amazon ECS. Les points de terminaison sont nécessaires dans les environnements de test (sandbox) dans lesquels aucune passerelle Internet (IGW) n'est connectée à Amazon VPC et aucune alternative ne mène à Internet. L'utilisation AWS PrivateLink garantit que les appels au service Amazon ECS restent dans le VPC Amazon et ne transitent pas par Internet. Pour savoir comment créer des AWS PrivateLink points de terminaison pour Amazon ECS et d'autres services connexes, consultez l'interface Amazon ECS et points de terminaison Amazon VPC.

> **Important**
>
> AWS Fargate les tâches ne nécessitent pas de point de AWS PrivateLink terminaison pour Amazon ECS.

Amazon ECR et Amazon ECS prennent tous deux en charge les stratégies de point de terminaison. Ces politiques vous permettent d'affiner l'accès à un service APIs. Par exemple, vous pouvez créer une stratégie de point de terminaison pour Amazon ECR qui autorise uniquement le transfert d'images vers des registres de comptes spécifiques AWS . Une telle stratégie pourrait être utilisée pour empêcher l'exfiltration de données par le biais d'images de conteneurs tout en permettant aux utilisateurs de les envoyer vers des registres Amazon ECR autorisés. Pour plus d'informations, veuillez consulter Utilisation des stratégies de point de terminaison pour contrôler l'accès à des points de terminaison d'un VPC.

La politique suivante permet à tous les AWS principaux de votre compte d'effectuer toutes les actions uniquement sur vos référentiels Amazon ECR :

```
{
  "Statement": [
    {
      "Sid": "LimitECRAccess",
      "Principal": "*",
      "Action": "*",
      "Effect": "Allow",
      "Resource": "arn:aws:ecr:region:account_id:repository/*"
    },
  ]
}
```

Vous pouvez encore améliorer cela en définissant une condition qui utilise la nouvelle propriété `PrincipalOrgID`. Cela empêche un directeur IAM qui ne fait pas partie de votre AWS Organizations entreprise de transférer et d'extraire des images. Pour plus d'informations, consultez aws : PrincipalOrg ID.

Nous avons recommandé d'appliquer la même politique aux points de terminaison `com.amazonaws.region.ecr.dkr` et `com.amazonaws.region.ecr.api`.

Paramètres de l'agent de conteneur

Le fichier de configuration de l'agent de conteneur Amazon ECS inclut plusieurs variables d'environnement liées à la sécurité du réseau. `ECS_AWSVPC_BLOCK_IMDS`et `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` sont utilisés pour bloquer l'accès d'une tâche aux EC2 métadonnées Amazon. `HTTP_PROXY`est utilisé pour configurer l'agent pour qu'il passe par un proxy HTTP afin de se connecter à Internet. Pour obtenir des instructions sur la configuration de l'agent et de l'environnement d'exécution Docker pour le routage via un proxy, veuillez consulter Configuration du proxy HTTP (langue française non garantie).

> **Important**
>
> Ces paramètres ne sont pas disponibles lorsque vous utilisez AWS Fargate.

Recommandations relatives à la sécurité du réseau

Nous vous recommandons de procéder comme suit lors de la configuration de votre Amazon VPC, de vos équilibreurs de charge et de votre réseau.

Utilisez le chiffrement réseau, le cas échéant, avec Amazon ECS

Vous devez utiliser le chiffrement réseau, le cas échéant. Certains programmes de conformité, tels que la norme PCI DSS, exigent que vous chiffriez les données en transit si celles-ci contiennent des données relatives au titulaire de la carte. Si votre charge de travail présente des exigences similaires, configurez le chiffrement réseau.

Les navigateurs modernes avertissent les utilisateurs lorsqu'ils se connectent à des sites non sécurisés. Si votre service est dirigé par un équilibreur de charge destiné au public, utilisez-le TLS/SSL pour chiffrer le trafic entre le navigateur du client et l'équilibreur de charge et pour le rechiffrer vers le backend si cela est justifié.

Utilisez le mode **`awsvpc`** réseau et les groupes de sécurité pour contrôler le trafic entre les tâches et les autres ressources dans Amazon ECS

Vous devez utiliser le mode réseau `awsvpc` et les groupes de sécurité lorsque vous devez contrôler le trafic entre les tâches et entre les tâches et les autres ressources du réseau. Si votre service se trouve derrière un ALB, utilisez des groupes de sécurité pour autoriser uniquement le trafic entrant

provenant d'autres ressources réseau utilisant le même groupe de sécurité que votre ALB. Si votre application se trouve derrière un NLB, configurez le groupe de sécurité de la tâche pour autoriser uniquement le trafic entrant provenant de la plage d'adresses CIDR Amazon VPC et des adresses IP statiques attribuées au NLB.

Les groupes de sécurité doivent également être utilisés pour contrôler le trafic entre les tâches et les autres ressources au sein d'Amazon VPC, telles que les bases de données Amazon RDS.

Créez des clusters Amazon ECS dans un Amazon distinct VPCs lorsque le trafic réseau doit être strictement isolé

Vous devez créer des clusters dans un Amazon distinct VPCs lorsque le trafic réseau doit être strictement isolé. Évitez d'exécuter des charges de travail soumises à des exigences de sécurité strictes sur des clusters dont les charges de travail ne sont pas tenues de respecter ces exigences. Lorsqu'une isolation réseau stricte est obligatoire, créez des clusters dans un Amazon distinct VPCs et exposez les services de manière sélective à d'autres Amazon à VPCs l'aide des points de terminaison Amazon VPC. Pour plus d'informations, consultez Points de terminaison d'un VPC.

Configurer les AWS PrivateLink points de terminaison lorsque cela est justifié pour Amazon ECS

Vous devez configurer les AWS PrivateLink points de terminaison lorsque cela est justifié. Si votre politique de sécurité vous empêche de connecter un Internet Gateway (IGW) à votre Amazon VPCs, configurez des AWS PrivateLink points de terminaison pour Amazon ECS et d'autres services tels qu'Amazon ECR AWS Secrets Manager et Amazon. CloudWatch

Utilisez Amazon VPC Flow Logs pour analyser le trafic à destination et en provenance de tâches de longue durée dans Amazon ECS

Vous devez utiliser les journaux de flux Amazon VPC pour analyser le trafic à destination et en provenance de tâches de longue durée. Les tâches qui utilisent le mode réseau `awsvpc` obtiennent leur propre ENI. Ainsi, vous pouvez surveiller le trafic à destination et en provenance de tâches individuelles à l'aide des journaux de flux Amazon VPC. Une récente mise à jour des journaux de flux Amazon VPC (v3) enrichit les journaux avec des métadonnées de trafic, notamment l'ID du VPC, l'ID de sous-réseau et l'ID d'instance. Ces métadonnées peuvent être utilisées pour affiner une enquête. Pour plus d'informations, veuillez consulter Journaux de flux Amazon VPC.

> **Note**
>
> En raison de la nature temporaire des conteneurs, les journaux de flux ne sont pas toujours un moyen efficace d'analyser les modèles de trafic entre les différents conteneurs ou entre les conteneurs et les autres ressources du réseau.

Bonnes pratiques en matière de sécurité des tâches et des conteneurs Amazon ECS

Vous devez considérer l'image du conteneur comme la première ligne de défense contre une attaque. Une image peu sécurisée et mal construite peut permettre à un attaquant d'échapper aux limites du conteneur et d'accéder à l'hôte. Vous devez procéder comme suit pour atténuer le risque que cela se produise.

Nous vous recommandons d'effectuer les opérations suivantes lorsque vous configurez vos tâches et conteneurs.

Créez un minimum d'images ou utilisez des images sans distribution

Commencez par retirer tous les fichiers binaires superflus de l'image de conteneur. Si vous utilisez une image inconnue provenant de la Galerie publique Amazon ECR, inspectez-la pour faire référence au contenu de chacune des couches du conteneur. Pour ce faire, vous pouvez utiliser une application telle que Dive.

Vous pouvez également utiliser des images sans distribution qui incluent uniquement votre application et ses dépendances d'exécution. Elles ne contiennent pas de gestionnaires de packages ou de shells. Les images sans distribution améliorent le « rapport signal/bruit émis par les scanners et réduisent la charge liée à l'établissement de la provenance à ce dont vous avez besoin ». Pour plus d'informations, consultez la GitHub documentation sur distroless.

Docker dispose d'un mécanisme permettant de créer des images à partir d'une image minimale réservée connue sous le nom de scratch. Pour plus d'informations, veuillez consulter Création d'une image parent simple à l'aide de scratch dans la documentation Docker (langue française non garantie). Avec des langages tels que Go, vous pouvez créer un fichier binaire lié statique et le référencer dans votre Dockerfile. L'exemple suivant montre comment procéder.

```
############################
```

```
# STEP 1 build executable binary
############################
FROM golang:alpine AS builder
# Install git.
# Git is required for fetching the dependencies.
RUN apk update && apk add --no-cache git
WORKDIR $GOPATH/src/mypackage/myapp/
COPY . .
# Fetch dependencies.
# Using go get.
RUN go get -d -v
# Build the binary.
RUN go build -o /go/bin/hello
############################
# STEP 2 build a small image
############################
FROM scratch
# Copy our static executable.
COPY --from=builder /go/bin/hello /go/bin/hello
# Run the hello binary.
ENTRYPOINT ["/go/bin/hello"]
This creates a container image that consists of your application and nothing else,
 making it extremely secure.
```

L'exemple suivant montre également un exemple de construction à plusieurs étapes. Ces types de constructions sont intéressants du point de vue de la sécurité, car vous pouvez les utiliser pour réduire la taille de l'image finale envoyée à votre registre de conteneurs. Les images de conteneur dépourvues d'outils de construction et d'autres fichiers binaires superflus améliorent votre niveau de sécurité en réduisant la surface d'attaque de l'image. Pour plus d'informations sur les versions en plusieurs étapes, voir les versions en plusieurs étapes dans la Docker documentation.

Analysez vos images pour détecter les vulnérabilités

À l'instar de leurs homologues machines virtuelles, les images de conteneur peuvent contenir des fichiers binaires et des bibliothèques d'applications présentant des vulnérabilités ou développer des vulnérabilités au fil du temps. Le meilleur moyen de se prémunir contre les attaques est d'analyser régulièrement vos images à l'aide d'un analyseur d'images.

Amazon ECR propose deux versions de l'analyse de base qui utilisent la base de données Common Vulnerabilities and Exposures (CVEs) :

- AWS numérisation de base native : utilise la technologie AWS native, qui est désormais GA et recommandée. Cette analyse de base améliorée est conçue pour fournir aux clients de meilleurs résultats d'analyse et une détection des vulnérabilités sur un large éventail de systèmes d'exploitation courants. Cela permet aux clients de renforcer encore la sécurité des images de leurs conteneurs. Tous les nouveaux registres de clients sont intégrés par défaut à cette version améliorée.
- Numérisation de base Clair — La version de numérisation de base précédente qui utilise le projet open source Clair et est obsolète. Pour plus d'informations sur Clair, voir Clair on GitHub.

La numérisation AWS native et la numérisation de base Clair sont prises en charge dans toutes les régions répertoriées dans AWS Services par région, à l'exception de celles ajoutées après septembre 2024. Le support de Clair étant obsolète, Clair ne sera pas pris en charge dans les nouvelles régions lors de leur ajout et ne sera plus pris en charge dans toutes les régions à compter du 1er octobre 2025.

Amazon ECR utilise la gravité d'un CVE à partir de la source de distribution en amont, si elle est disponible. Dans le cas contraire, le score CVSS (Common Vulnerability Scoring System) est utilisé. Le score CVSS peut être utilisé pour obtenir l'évaluation de gravité de la vulnérabilité NVD. Pour en savoir plus, consultez NVD Vulnerability Gravity Ratings.

Vous pouvez également consulter les résultats d'un scan depuis la console Amazon ECR ou en appelant l'DescribeImageScanFindingsAPI. Les images présentant une vulnérabilité `HIGH` ou `CRITICAL` doivent être supprimées ou reconstruites. Si une image déployée développe une vulnérabilité, elle doit être remplacée dès que possible.

Docker Desktop Edge version 2.3.6.0 ou ultérieure peut analyser des images locales. Les analyses sont fournies par Snyk, un service de sécurité des applications. Lorsque des vulnérabilités sont découvertes, Snyk identifie les couches et les dépendances contenant la vulnérabilité dans le Dockerfile. Il recommande également des alternatives sûres, telles que l'utilisation d'une image de base plus fine présentant moins de vulnérabilités ou la mise à niveau d'un package particulier vers une version plus récente. En utilisant l'analyse Docker, les développeurs peuvent résoudre les problèmes de sécurité potentiels avant de transférer leurs images dans le registre.

- Le billet de blog Automating image compliance using Amazon ECR and AWS Security Hub explique comment intégrer les informations relatives aux vulnérabilités issues d'Amazon ECR dans AWS Security Hub et automatiser la correction en bloquant l'accès aux images vulnérables.

Supprimez les autorisations spéciales associées à vos images

Les indicateurs des droits d'accès `setuid` et `setgid` autorisent l'exécution d'un exécutable avec les autorisations du propriétaire ou du groupe de l'exécutable. Supprimez de votre image tous les fichiers binaires dotés de ces droits d'accès, car ils peuvent être utilisés pour escalader les privilèges. Envisagez de supprimer tous les shells et utilitaires comme `nc` ou `curl` susceptibles d'être utilisés à des fins malveillantes. Vous pouvez trouver les fichiers avec les droits d'accès `setuid` et `setgid` à l'aide de la commande suivante.

```
find / -perm /6000 -type f -exec ls -ld {} \;
```

Pour supprimer ces autorisations spéciales de ces fichiers, ajoutez la directive suivante à votre image de conteneur.

```
RUN find / -xdev -perm /6000 -type f -exec chmod a-s {} \; || true
```

Créez un ensemble d'images sélectionnées

Plutôt que de permettre aux développeurs de créer leurs propres images, créez un ensemble d'images validées pour les différentes piles d'applications de votre organisation. Ce faisant, les développeurs peuvent renoncer à apprendre à composer des Dockerfiles et se concentrer sur l'écriture de code. Au fur et à mesure que les modifications sont fusionnées dans votre base de code, un pipeline CI/CD peut automatiquement compiler la ressource, puis la stocker dans un référentiel d'artefacts. Enfin, copiez l'artefact dans l'image appropriée avant de le transférer vers un registre Docker tel qu'Amazon ECR. Vous devez créer au minimum un ensemble d'images de base à partir desquelles les développeurs peuvent créer leurs propres Dockerfiles. Vous devez éviter d'extraire des images depuis Docker Hub. Vous ne savez pas toujours ce que contient l'image : environ un cinquième des 1 000 images principales présente des vulnérabilités. Une liste de ces images et de leurs vulnérabilités est disponible sur https://vulnerablecontainers.org/.

Analysez les packages d'applications et les bibliothèques pour détecter les vulnérabilités

L'utilisation de bibliothèques open source est désormais courante. Comme les systèmes d'exploitation et les packages de systèmes d'exploitation, ces bibliothèques peuvent présenter des

vulnérabilités. Dans le cadre du cycle de développement, ces bibliothèques doivent être analysées et mises à jour lorsque des vulnérabilités critiques sont détectées.

Docker Desktop effectue des analyses locales à l'aide de Snyk. Il peut également être utilisé pour détecter des vulnérabilités et des problèmes de licence potentiels dans les bibliothèques open source. Il peut être intégré directement dans les flux de travail des développeurs, ce qui vous permet d'atténuer les risques posés par les bibliothèques open source. Pour plus d'informations, consultez les rubriques suivantes :

- La page outils de sécurité des applications open source inclut une liste d'outils permettant de détecter les vulnérabilités des applications.

Effectuez une analyse de code statique

Vous devez effectuer une analyse de code statique avant de créer une image de conteneur. Celle-ci est effectuée sur votre code source et est utilisée pour identifier les erreurs de codage et le code susceptible d'être utilisé par un acteur malveillant, comme les injections d'erreurs. Vous pouvez utiliser Amazon Inspector. Pour plus d'informations, consultez la section Numérisation d'images de conteneurs Amazon ECR avec Amazon Inspector dans le guide de l'utilisateur d'Amazon Inspector.

Exécutez les conteneurs en tant qu'utilisateur non root

Vous devez exécuter les conteneurs en tant qu'utilisateur non root. Par défaut, les conteneurs s'exécutent en tant qu'utilisateur `root`, sauf si la directive `USER` est incluse dans votre Dockerfile. Les fonctionnalités Linux par défaut attribuées par Docker limitent les actions qui peuvent être exécutées en tant que `root`, mais seulement de manière marginale. Par exemple, un conteneur s'exécutant en tant que `root` n'est toujours pas autorisé à accéder aux appareils.

Dans le cadre de votre CI/CD pipeline, vous devez lancer Dockerfiles pour rechercher la `USER` directive et échouer à la compilation si elle est manquante. Pour plus d'informations, consultez les rubriques suivantes :

- Dockerfile-lint est un outil open source RedHat qui peut être utilisé pour vérifier si le fichier est conforme aux meilleures pratiques.
- Hadolint est un autre outil permettant de créer des images Docker conformes aux bonnes pratiques.

Utilisez un système de fichiers racine en lecture seule

Vous devez utiliser un système de fichiers racine en lecture seule. Le système de fichiers racine d'un conteneur est accessible en écriture par défaut. Lorsque vous configurez un conteneur avec un système de fichiers racine `RO` (en lecture seule), cela vous oblige à définir explicitement où les données peuvent être conservées. Cela réduit votre surface d'attaque, car le système de fichiers du conteneur ne peut pas être disponible en écriture à moins que des autorisations ne soient spécifiquement accordées.

> **Note**
>
> Le fait d'avoir un système de fichiers racine en lecture seule peut entraîner des problèmes avec certains packages du système d'exploitation qui devraient pouvoir écrire dans le système de fichiers. Si vous prévoyez d'utiliser des systèmes de fichiers racine en lecture seule, effectuez des tests approfondis au préalable.

Configuration des tâches avec des limites de processeur et de mémoire (Amazon EC2)

Vous devez configurer les tâches avec des limites de processeur et de mémoire afin de réduire les risques suivants. Les limites de ressources d'une tâche définissent une limite supérieure pour la quantité de processeur et de mémoire pouvant être réservée par tous les conteneurs d'une tâche. Si aucune limite n'est définie, les tâches ont accès au processeur et à la mémoire de l'hôte. Cela peut entraîner des problèmes, car les tâches déployées sur un hôte partagé peuvent priver d'autres tâches de ressources système.

> **Note**
>
> Amazon ECS sur les AWS Fargate tâches vous oblige à spécifier des limites de processeur et de mémoire, car il utilise ces valeurs à des fins de facturation. Une seule tâche monopolisant toutes les ressources du système n'est pas un problème pour Amazon ECS Fargate, car chaque tâche est exécutée sur sa propre instance dédiée. Si vous ne spécifiez pas de limite de mémoire, Amazon ECS alloue un minimum de 4 Mo à chaque conteneur. De même, si aucune limite de processeur n'est définie pour la tâche, l'agent de conteneur Amazon ECS lui en attribue un minimum de 2 CPUs.

Utilisez des balises immuables avec Amazon ECR

Avec Amazon ECR, vous pouvez et devez utiliser des images configurées avec des balises immuables. Cela empêche de transférer une version modifiée ou mise à jour d'une image vers votre référentiel d'images avec une balise identique. Cela permet d'éviter qu'un attaquant ne diffuse une version compromise d'une image sur votre image portant la même balise. En utilisant des balises immuables, vous vous forcez efficacement à envoyer une nouvelle image avec une balise différente pour chaque modification.

Évitez de gérer les conteneurs en tant que privilégiés (Amazon EC2)

Vous devez éviter d'exécuter des conteneurs dotés de privilèges. En effet, les conteneurs s'exécutant comme `privileged` sont exécutés avec des privilèges étendus sur l'hôte. Cela signifie que le conteneur hérite de toutes les fonctionnalités Linux attribuées à `root` sur l'hôte. Son utilisation doit être strictement restreinte ou interdite. Nous vous conseillons de définir la variable d'environnement de l'agent de conteneur Amazon ECS `ECS_DISABLE_PRIVILEGED` sur `true` afin d'empêcher les conteneurs de s'exécuter comme `privileged` sur des hôtes particuliers si `privileged` n'est pas nécessaire. Vous pouvez également AWS Lambda analyser vos définitions de tâches afin d'utiliser le `privileged` paramètre.

> **Note**
>
> L'exécution d'un conteneur en tant que `privileged` n'est pas prise en charge sur Amazon ECS sur AWS Fargate.

Supprimez les fonctionnalités Linux inutiles du conteneur

Voici une liste des fonctionnalités Linux par défaut attribuées aux conteneurs Docker. Pour plus d'informations sur chaque fonctionnalité, veuillez consulter Présentation des fonctionnalités de Linux (langue française non garantie).

```
CAP_CHOWN, CAP_DAC_OVERRIDE, CAP_FOWNER, CAP_FSETID, CAP_KILL,
CAP_SETGID, CAP_SETUID, CAP_SETPCAP, CAP_NET_BIND_SERVICE,
CAP_NET_RAW, CAP_SYS_CHROOT, CAP_MKNOD, CAP_AUDIT_WRITE,
CAP_SETFCAP
```

Si un conteneur ne nécessite pas toutes les fonctionnalités de noyau Docker répertoriées ci-dessus, pensez à les supprimer du conteneur. Pour plus d'informations sur les fonctionnalités de chaque noyau Docker, consultez KernelCapabilities. Vous pouvez déterminer les fonctionnalités utilisées en procédant ainsi :

- Installez le package du système d'exploitation libcap-ng et exécutez l'utilitaire `pscap` pour répertorier les fonctionnalités utilisées par chaque processus.
- Vous pouvez également utiliser capsh pour déchiffrer les fonctionnalités utilisées par un processus.

Utilisez une clé gérée par le client (CMK) pour chiffrer les images transférées vers Amazon ECR

Vous devez utiliser une clé gérée par le client (CMK) pour chiffrer les images transférées vers Amazon ECR. Les images transmises à Amazon ECR sont automatiquement chiffrées au repos à l'aide d'une clé gérée AWS Key Management Service (AWS KMS). Si vous préférez utiliser votre propre clé, Amazon ECR prend désormais en charge le AWS KMS chiffrement avec des clés gérées par le client (CMK). Avant d'activer le chiffrement côté serveur avec une clé CMK, consultez les considérations répertoriées dans la documentation sur le chiffrement au repos.

Didacticiels pour Amazon ECS

Les didacticiels suivants montrent comment exécuter les tâches courantes à l'aide d'Amazon ECS.

Vous pouvez utiliser l'un des didacticiels suivants pour en savoir plus sur la prise en main d'Amazon ECS.

Aperçu du didacticiel	En savoir plus	
Commencez à utiliser Amazon ECS sur Fargate.	Découvrez comment créer une tâche Linux Amazon ECS pour le type de lancement Fargate	
Commencez à utiliser les conteneurs Windows sur Fargate.	Découvrez comment créer une tâche Windows Amazon ECS pour le type de lancement Fargate	
Commencez avec les conteneurs Windows pour le type de EC2 lancement.	Découvrez comment créer une tâche Windows Amazon ECS pour le type de EC2 lancement	

Vous pouvez utiliser l'un des didacticiels suivants pour déployer des tâches sur Amazon ECS à l'aide du AWS CLI

Aperçu du didacticiel	En savoir plus	
Créez une tâche Linux pour le type de lancement Fargate.	Création d'une tâche Linux Amazon ECS pour le type de lancement Fargate avec AWS CLI	
Créez une tâche Windows pour le type de lancement Fargate.	Création d'une tâche Windows Amazon ECS pour le type de	

Aperçu du didacticiel	En savoir plus	
	lancement Fargate avec AWS CLI	
Créez une tâche Linux pour le type de EC2 lancement.	Création d'une tâche Amazon ECS pour le type de EC2 lancement avec AWS CLI	

Vous pouvez utiliser l'un des didacticiels suivants pour en savoir plus sur la surveillance et la journalisation.

Aperçu du didacticiel	En savoir plus	
Configurez une fonction Lambda simple qui écoute les événements des tâches et les écrit dans un flux de CloudWatch journal des journaux.	Configuration d'Amazon ECS pour écouter CloudWatch les événements	
Configurez une règle d' EventBridge événement Amazon qui capture uniquement les événement s de tâche lorsque la tâche a cessé de s'exécuter parce que l'un de ses conteneurs essentiels s'est arrêté.	Envoi d'alertes Amazon Simple Notification Service pour les événements liés à l'arrêt d'une tâche Amazon ECS	
Concaténez les messages de journal qui appartenaient à l'origine à un contexte mais qui étaient répartis sur plusieurs enregistrements ou lignes de journal.	Concaténation de messages de journal Amazon ECS multilignes ou empilés	

Aperçu du didacticiel	En savoir plus	
Déployez des conteneurs Fluent Bit sur leurs instances Windows exécutées dans Amazon ECS pour diffuser les journaux générés par les tâches Windows vers Amazon afin CloudWatch de centraliser la journalisation.	Déploiement de Fluent Bit sur des conteneurs Windows Amazon ECS	

Vous pouvez utiliser l'un des didacticiels suivants pour en savoir plus sur l'utilisation de l'authentification Active Directory avec un compte de service géré de groupe sur Amazon ECS.

Aperçu du didacticiel	En savoir plus	
Utilisez un compte de service géré de groupe avec des conteneurs Linux activés EC2.	Utilisation gMSA pour les EC2 Linux conteneurs sur Amazon ECS	
Utilisez un compte de service géré de groupe avec des conteneurs Windows activés EC2.	Découvrez comment utiliser les conteneurs g MSAs pour EC2 Windows pour Amazon ECS	
Utilisez un compte de service géré de groupe avec des conteneurs Linux sur Fargate.	Utilisation gMSA pour les Linux conteneurs sur Fargate	
Créez une tâche qui exécute un conteneur Windows doté des informations d'identif ication permettant d'accéder à Active Directory avec un compte de service géré de groupe sans domaine.	Utilisation de conteneurs Windows Amazon ECS avec des conteneurs sans domaine à gMSA l'aide du AWS CLI	

Création d'une tâche Linux Amazon ECS pour le type de lancement Fargate avec AWS CLI

Les étapes suivantes vous aideront à configurer un cluster, enregistrer une définition de tâche, exécuter une tâche Linux et effectuer d'autres scénarios courants dans Amazon ECS avec la AWS CLI. Utilisez la dernière version du AWS CLI. Pour plus d'informations sur la mise à niveau vers la dernière version, voir Installation ou mise à jour vers la dernière version du AWS CLI.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir de AWS CLI SDKs, et de l'API Amazon ECS via les deux IPv4 et. IPv6 Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

Rubriques

- Prérequis
- Étape 1 : Créer un cluster
- Étape 2 : Enregistrer une définition de tâche Linux
- Étape 3 : Répertorier les définitions de tâche
- Étape 4 : Créer un service
- Étape 5 : Répertorier les services
- Étape 6 : Décrire le service en cours d'exécution
- Étape 7 : Test
- Étape 8 : Nettoyer

Prérequis

Le didacticiel suppose de remplir les prérequis suivants.

- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations sur l'installation ou la mise à niveau de votre AWS CLI, Installation ou mise à jour vers la dernière version du AWS CLI.

- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique Amazon ECS_ FullAccess IAM.
- Vous avez un créé un VPC et un groupe de sécurité prêts à être utilisés. Ce didacticiel utilise une image de conteneur hébergée sur Amazon ECR Public afin que votre tâche ait accès à Internet. Pour donner à votre tâche un itinéraire vers Internet, utilisez l'une des options suivantes.
  - Utilisez un sous-réseau privé avec une passerelle NAT dotée d'une adresse IP Elastic.
  - Utilisez un sous-réseau public et affectez une adresse IP publique à la tâche.

  Pour de plus amples informations, veuillez consulter the section called “Créer un Virtual Private Cloud”.

  Pour plus d'informations sur les groupes de sécurité et les règles, consultez les sections Groupes de sécurité par défaut pour vous VPCs et Exemples de règles dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.
- Si vous suivez ce didacticiel en utilisant un sous-réseau privé, vous pouvez utiliser Amazon ECS Exec pour interagir directement avec votre conteneur et tester le déploiement. Vous devez créer un rôle IAM de tâche pour utiliser ECS Exec. Pour plus d'informations sur le rôle IAM de la tâche et les autres prérequis, consultez Surveiller les conteneurs Amazon ECS avec Amazon ECS Exec.
- (Facultatif) AWS CloudShell est un outil qui fournit aux clients une ligne de commande sans avoir à créer leur propre EC2 instance. Pour plus d'informations, voir Qu'est-ce que c'est AWS CloudShell ? dans le guide de AWS CloudShell l'utilisateur.

Étape 1 : Créer un cluster

Par défaut, votre compte reçoit un cluster `default`.

> **Note**
>
> L'utilisation du cluster `default` qui vous est fourni présente l'avantage que vous n'avez pas besoin de spécifier l'option `--cluster` *`cluster_name`* dans les commandes suivantes. Si vous créez votre propre cluster, sans valeur par défaut, vous devez spécifier `--cluster` *`cluster_name`* pour chaque commande que vous prévoyez d'utiliser avec ce cluster.

Créez votre propre cluster avec un nom unique à l'aide de la commande suivante :

```
aws ecs create-cluster --cluster-name fargate-cluster
```

Sortie :

```
{
    "cluster": {
        "status": "ACTIVE",
        "defaultCapacityProviderStrategy": [],
        "statistics": [],
        "capacityProviders": [],
        "tags": [],
        "clusterName": "fargate-cluster",
        "settings": [
            {
                "name": "containerInsights",
                "value": "disabled"
            }
        ],
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster"
    }
}
```

Étape 2 : Enregistrer une définition de tâche Linux

Avant de pouvoir exécuter une tâche sur votre cluster ECS, vous devez enregistrer une définition de tâche. Les définitions de tâches sont des listes de conteneurs regroupés ensemble. L'exemple suivant est une définition de tâche simple qui crée une application Web PHP à l'aide de l'image de conteneur httpd hébergée sur Docker Hub. Pour plus d'informations sur les paramètres de définition des tâches disponibles, consultez Définitions de tâche Amazon ECS. Pour ce didacticiel, l'`taskRoleArn` n'est nécessaire que si vous déployez la tâche dans un sous-réseau privé et souhaitez tester le déploiement. Remplacez l'`taskRoleArn` par le rôle de tâche IAM que vous avez créé pour utiliser ECS Exec, comme indiqué dans Prérequis.

```
 {
        "family": "sample-fargate",
        "networkMode": "awsvpc",
```

```
        "taskRoleArn": "arn:aws:iam::aws_account_id:role/execCommandRole",
        "containerDefinitions": [
            {
                "name": "fargate-app",
                "image": "public.ecr.aws/docker/library/httpd:latest",
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80,
                        "protocol": "tcp"
                    }
                ],
                "essential": true,
                "entryPoint": [
                    "sh",
                    "-c"
                ],
                "command": [
                    "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</h1>
 <h2>Congratulations!</h2> <p>Your application is now running on a container in Amazon
 ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html && httpd-
foreground\""
                ]
            }
        ],
        "requiresCompatibilities": [
            "FARGATE"
        ],
        "cpu": "256",
        "memory": "512"
}
```

Enregistrez la définition de tâche JSON sous forme de fichier et transmettez-la avec l'option `--cli-input-json file://path_to_file.json`.

Pour utiliser un fichier JSON pour les définitions de conteneur :

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/fargate-task.json
```

La commande register-task-definition renvoie une description de la définition de tâche une fois l'enregistrement terminé.

Étape 3 : Répertorier les définitions de tâche

Vous pouvez répertorier les définitions de tâches de votre compte à tout moment à l'aide de la commande list-task-definitions. La sortie de cette commande affiche les valeurs `family` et `revision` que vous pouvez utiliser ensemble lorsque vous appelez run-task ou start-task.

```
aws ecs list-task-definitions
```

Sortie :

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:region:aws_account_id:task-definition/sample-fargate:1"
    ]
}
```

Étape 4 : Créer un service

Une fois que vous avez enregistré une tâche pour votre compte, vous pouvez créer un service pour la tâche enregistrée dans votre cluster. Pour cet exemple, vous créez un service avec une instance de la définition de tâche `sample-fargate:1` exécutée dans votre cluster. La tâche nécessite un itinéraire vers Internet. Il y a deux façons d'y parvenir. Une façon consiste à utiliser un sous-réseau privé configuré avec une passerelle NAT avec une adresse IP Elastic dans un sous-réseau public. Une autre façon consiste à utiliser un sous-réseau public et à attribuer une adresse IP publique à votre tâche. Les deux exemples sont présentés ci-dessous.

Exemple d'utilisation d'un sous-réseau privé. L'option `enable-execute-command` est nécessaire pour utiliser Amazon ECS Exec.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service --
task-definition sample-fargate:1 --desired-count 1 --launch-type "FARGATE" --network-
configuration "awsvpcConfiguration={subnets=[subnet-abcd1234],securityGroups=[sg-
abcd1234]}" --enable-execute-command
```

Exemple d'utilisation d'un sous-réseau public.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service --
task-definition sample-fargate:1 --desired-count 1 --launch-type "FARGATE" --network-
```

```
configuration "awsvpcConfiguration={subnets=[subnet-abcd1234],securityGroups=[sg-abcd1234],assignPublicIp=ENABLED}"
```

La commande create-service renvoie une description de la définition de tâche une fois l'enregistrement terminé.

Étape 5 : Répertorier les services

Affichez les services de votre cluster. Vous devriez voir le service que vous avez exécuté dans la section précédente. Vous pouvez noter le nom du service ou l'ARN complet qui est renvoyé par cette commande et l'utiliser pour décrire le service ultérieurement.

```
aws ecs list-services --cluster fargate-cluster
```

Sortie :

```
{
    "serviceArns": [
        "arn:aws:ecs:region:aws_account_id:service/fargate-cluster/fargate-service"
    ]
}
```

Étape 6 : Décrire le service en cours d'exécution

Décrivez le service à l'aide du nom de service récupéré précédemment afin d'obtenir plus d'informations sur la tâche.

```
aws ecs describe-services --cluster fargate-cluster --services fargate-service
```

En cas de réussite, une description des défaillances de service et des services est renvoyée. Par exemple, dans la section `services`, vous trouverez des informations sur les déploiements, telles que le statut des tâches en cours d'exécution ou en attente. Vous trouverez également des informations sur la définition des tâches, la configuration réseau et les événements horodatés. Dans la section des défaillances, vous trouverez des informations sur les défaillances, le cas échéant, associées à l'appel. Pour le dépannage, consultez Messages d'événements de service. Pour de plus amples informations sur la description du service, consultez Description des services.

```
{
    "services": [
        {
```

```
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-abcd1234"
                    ],
                    "securityGroups": [
                        "sg-abcd1234"
                    ],
                    "assignPublicIp": "ENABLED"
                }
            },
            "launchType": "FARGATE",
            "enableECSManagedTags": false,
            "loadBalancers": [],
            "deploymentController": {
                "type": "ECS"
            },
            "desiredCount": 1,
            "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster",
            "serviceArn": "arn:aws:ecs:region:aws_account_id:service/fargate-service",
            "deploymentConfiguration": {
                "maximumPercent": 200,
                "minimumHealthyPercent": 100
            },
            "createdAt": 1692283199.771,
            "schedulingStrategy": "REPLICA",
            "placementConstraints": [],
            "deployments": [
                {
                    "status": "PRIMARY",
                    "networkConfiguration": {
                        "awsvpcConfiguration": {
                            "subnets": [
                                "subnet-abcd1234"
                            ],
                            "securityGroups": [
                                "sg-abcd1234"
                            ],
                            "assignPublicIp": "ENABLED"
                        }
                    },
                    "pendingCount": 0,
                    "launchType": "FARGATE",
                    "createdAt": 1692283199.771,
```

```
                    "desiredCount": 1,
                    "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-
definition/sample-fargate:1",
                    "updatedAt": 1692283199.771,
                    "platformVersion": "1.4.0",
                    "id": "ecs-svc/9223370526043414679",
                    "runningCount": 0
                }
            ],
            "serviceName": "fargate-service",
            "events": [
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 53c0de40-ea3b-489f-a352-623bf1235f08) (task d0aec985-901b-488f-9fb4-61b991b332a3).",
                    "id": "92b8443e-67fb-4886-880c-07e73383ea83",
                    "createdAt": 1510811841.408
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 b4911bee-7203-4113-99d4-e89ba457c626) (task cc5853e3-6e2d-4678-8312-74f8a7d76474).",
                    "id": "d85c6ec6-a693-43b3-904a-a997e1fc844d",
                    "createdAt": 1510811601.938
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 cba86182-52bf-42d7-9df8-b744699e6cfc) (task f4c1ad74-a5c6-4620-90cf-2aff118df5fc).",
                    "id": "095703e1-0ca3-4379-a7c8-c0f1b8b95ace",
                    "createdAt": 1510811364.691
                }
            ],
            "runningCount": 0,
            "status": "ACTIVE",
            "serviceRegistries": [],
            "pendingCount": 0,
            "createdBy": "arn:aws:iam::aws_account_id:user/user_name",
            "platformVersion": "LATEST",
            "placementStrategy": [],
            "propagateTags": "NONE",
            "roleArn": "arn:aws:iam::aws_account_id:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
            "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-definition/
sample-fargate:1"
        }
    ],
```

```
    "failures": []
}
```

Étape 7 : Test

Tâche de test déployée à l'aide d'un sous-réseau public

Décrivez la tâche dans le service afin que vous puissiez obtenir l'interface réseau Elastic (ENI) pour la tâche.

Tout d'abord, obtenez l'ARN de la tâche.

```
aws ecs list-tasks --cluster fargate-cluster --service fargate-service
```

La sortie contient l'ARN de la tâche.

```
{
    "taskArns": [
        "arn:aws:ecs:us-east-1:123456789012:task/fargate-service/EXAMPLE
    ]
}
```

Décrivez la tâche et localisez l'ID de l'ENI. Utilisez l'ARN de la tâche pour le paramètre `tasks`.

```
aws ecs describe-tasks --cluster fargate-cluster --tasks arn:aws:ecs:us-
east-1:123456789012:task/service/EXAMPLE
```

Les informations concernant la pièce jointe sont répertoriées dans la sortie.

```
{
    "tasks": [
        {
            "attachments": [
                {
                    "id": "d9e7735a-16aa-4128-bc7a-b2d5115029e9",
                    "type": "ElasticNetworkInterface",
                    "status": "ATTACHED",
                    "details": [
                        {
                            "name": "subnetId",
                            "value": "subnetabcd1234"
                        },
```

```
                        {
                            "name": "networkInterfaceId",
                            "value": "eni-0fa40520aeEXAMPLE"
                        },
                    ]
                }
…
}
```

Décrivez l'ENI pour obtenir l'adresse IP publique.

```
aws ec2 describe-network-interfaces --network-interface-id  eni-0fa40520aeEXAMPLE
```

L'adresse IP publique se trouve dans la sortie.

```
{
    "NetworkInterfaces": [
        {
            "Association": {
                "IpOwnerId": "amazon",
                "PublicDnsName": "ec2-34-229-42-222.compute-1.amazonaws.com",
                "PublicIp": "198.51.100.2"
            },
…
}
```

Saisissez l'adresse IP publique dans votre navigateur web. Vous devriez voir une page web qui affiche l'exemple d'application Amazon ECS.

Tâche de test déployée à l'aide d'un sous-réseau privé

Décrivez la tâche et localisez `managedAgents` pour vérifier que `ExecuteCommandAgent` est en cours d'exécution. Notez `privateIPv4Address` pour une utilisation ultérieure.

```
aws ecs describe-tasks --cluster fargate-cluster --tasks arn:aws:ecs:us-
east-1:123456789012:task/fargate-service/EXAMPLE
```

Les informations concernant l'agent géré sont répertoriées dans la sortie.

```
{
     "tasks": [
        {
```

```
            "attachments": [
                {
                    "id": "d9e7735a-16aa-4128-bc7a-b2d5115029e9",
                    "type": "ElasticNetworkInterface",
                    "status": "ATTACHED",
                    "details": [
                        {
                            "name": "subnetId",
                            "value": "subnetabcd1234"
                        },
                        {
                            "name": "networkInterfaceId",
                            "value": "eni-0fa40520aeEXAMPLE"
                        },
                        {
                            "name": "privateIPv4Address",
                            "value": "10.0.143.156"
                        }
                    ]
                }
            ],
    ...
    "containers": [
        {
        ...
      "managedAgents": [
                        {
                            "lastStartedAt": "2023-08-01T16:10:13.002000+00:00",
                            "name": "ExecuteCommandAgent",
                            "lastStatus": "RUNNING"
                        }
                ],
      ...
  }
```

Après avoir vérifié que `ExecuteCommandAgent` est en cours d'exécution, vous pouvez exécuter la commande suivante pour exécuter un shell interactif sur le conteneur de la tâche.

```
aws ecs execute-command --cluster fargate-cluster \
    --task  arn:aws:ecs:us-east-1:123456789012:task/fargate-service/EXAMPLE  \
    --container  fargate-app \
    --interactive \
    --command "/bin/sh"
```

Une fois le shell interactif lancé, exécutez les commandes suivantes pour installer cURL.

```
apt update
```

```
apt install curl
```

Après avoir installé cURL, exécutez la commande suivante en utilisant l'adresse IP privée que vous avez obtenue précédemment.

```
 curl 10.0.143.156
```

Vous devriez voir l'équivalent HTML de la page Web d'exemple d'application Amazon ECS.

```
<html>
    <head>
     <title>Amazon ECS Sample App</title>
     <style>body {margin-top: 40px; background-color: #333;} </style>
    </head>
      <body>
      <div style=color:white;text-align:center>
      <h1>Amazon ECS Sample App</h1>
      <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p>
      </div>
      </body>
</html>
```

Étape 8 : Nettoyer

Une fois que vous avez terminé ce didacticiel, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources inutilisées.

Supprimez le service.

```
aws ecs delete-service --cluster fargate-cluster --service fargate-service --force
```

Supprimez le cluster.

```
aws ecs delete-cluster --cluster fargate-cluster
```

Création d'une tâche Windows Amazon ECS pour le type de lancement Fargate avec AWS CLI

Les étapes suivantes vous aideront à configurer un cluster, enregistrer une définition de tâche, exécuter une tâche Windows et effectuer d'autres scénarios courants dans Amazon ECS avec la AWS CLI. Veillez à utiliser la dernière version de l' AWS CLI. Pour plus d'informations sur la mise à niveau vers la dernière version, voir Installation ou mise à jour vers la dernière version du AWS CLI.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir de AWS CLI SDKs, et de l'API Amazon ECS via les deux IPv4 et. IPv6 Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

Rubriques

- Prérequis
- Étape 1 : Créer un cluster
- Étape 2 : Enregistrement d'une définition de tâche Windows
- Étape 3 : Répertorier les définitions de tâche
- Étape 4 : Créer un service
- Étape 5 : Répertorier les services
- Étape 6 : Décrire le service en cours d'exécution
- Étape 7 : Nettoyer

Prérequis

Le didacticiel suppose de remplir les prérequis suivants.

- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations sur l'installation ou la mise à niveau de votre AWS CLI AWS CLI.
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique Amazon ECS_ FullAccess IAM.

- Vous avez un créé un VPC et un groupe de sécurité prêts à être utilisés. Ce didacticiel utilise une image de conteneur hébergée sur Docker Hub afin que votre tâche ait accès à Internet. Pour donner à votre tâche un itinéraire vers Internet, utilisez l'une des options suivantes.
  - Utilisez un sous-réseau privé avec une passerelle NAT dotée d'une adresse IP Elastic.
  - Utilisez un sous-réseau public et affectez une adresse IP publique à la tâche.

  Pour de plus amples informations, veuillez consulter the section called “Créer un Virtual Private Cloud”.

  Pour plus d'informations sur les groupes de sécurité et les règles, consultez les sections Groupes de sécurité par défaut pour vous VPCs et Exemples de règles dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.
- (Facultatif) AWS CloudShell est un outil qui fournit aux clients une ligne de commande sans avoir à créer leur propre EC2 instance. Pour plus d'informations, voir Qu'est-ce que c'est AWS CloudShell ? dans le guide de AWS CloudShell l'utilisateur.

Étape 1 : Créer un cluster

Par défaut, votre compte reçoit un cluster `default`.

> **Note**
>
> L'utilisation du cluster `default` qui vous est fourni présente l'avantage que vous n'avez pas besoin de spécifier l'option `--cluster` *`cluster_name`* dans les commandes suivantes. Si vous créez votre propre cluster, sans valeur par défaut, vous devez spécifier `--cluster` *`cluster_name`* pour chaque commande que vous prévoyez d'utiliser avec ce cluster.

Créez votre propre cluster avec un nom unique à l'aide de la commande suivante :

```
aws ecs create-cluster --cluster-name fargate-cluster
```

Sortie :

```
{
    "cluster": {
        "status": "ACTIVE",
```

```
        "statistics": [],
        "clusterName": "fargate-cluster",
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster"
    }
}
```

Étape 2 : Enregistrement d'une définition de tâche Windows

Avant de pouvoir exécuter une tâche Windows sur votre cluster Amazon ECS, vous devez enregistrer une définition de tâche. Les définitions de tâches sont des listes de conteneurs regroupés ensemble. L'exemple suivant est une simple définition de tâche qui crée une application web. Pour plus d'informations sur les paramètres de définition des tâches disponibles, consultez Définitions de tâche Amazon ECS.

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
```

```
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

L'exemple JSON ci-dessus peut être transmis AWS CLI au de deux manières : vous pouvez enregistrer la définition de tâche JSON sous forme de fichier et la transmettre avec l'`--cli-input-json file://`*`path_to_file.json`*option.

Pour utiliser un fichier JSON pour les définitions de conteneur :

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/fargate-task.json
```

La commande register-task-definition renvoie une description de la définition de tâche une fois l'enregistrement terminé.

Étape 3 : Répertorier les définitions de tâche

Vous pouvez répertorier les définitions de tâches de votre compte à tout moment à l'aide de la commande list-task-definitions. La sortie de cette commande affiche les valeurs `family` et `revision` que vous pouvez utiliser ensemble lorsque vous appelez run-task ou start-task.

```
aws ecs list-task-definitions
```

Sortie :

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:region:aws_account_id:task-definition/sample-fargate-windows:1"
    ]
}
```

Étape 4 : Créer un service

Une fois que vous avez enregistré une tâche pour votre compte, vous pouvez créer un service pour la tâche enregistrée dans votre cluster. Pour cet exemple, vous créez un service avec une instance de la définition de tâche `sample-fargate:1` exécutée dans votre cluster. La tâche nécessite un itinéraire vers Internet. Il y a deux façons d'y parvenir. Une façon consiste à utiliser un sous-réseau privé configuré avec une passerelle NAT avec une adresse IP Elastic dans un sous-réseau public. Une autre façon consiste à utiliser un sous-réseau public et à attribuer une adresse IP publique à votre tâche. Les deux exemples sont présentés ci-dessous.

Exemple d'utilisation d'un sous-réseau privé.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service
 --task-definition sample-fargate-windows:1 --desired-count 1 --launch-type
 "FARGATE" --network-configuration "awsvpcConfiguration={subnets=[subnet-
abcd1234],securityGroups=[sg-abcd1234]}"
```

Exemple d'utilisation d'un sous-réseau public.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service
 --task-definition sample-fargate-windows:1 --desired-count 1 --launch-type
 "FARGATE" --network-configuration "awsvpcConfiguration={subnets=[subnet-
abcd1234],securityGroups=[sg-abcd1234],assignPublicIp=ENABLED}"
```

La commande create-service renvoie une description de la définition de tâche une fois l'enregistrement terminé.

Étape 5 : Répertorier les services

Affichez les services de votre cluster. Vous devriez voir le service que vous avez exécuté dans la section précédente. Vous pouvez noter le nom du service ou l'ARN complet qui est renvoyé par cette commande et l'utiliser pour décrire le service ultérieurement.

```
aws ecs list-services --cluster fargate-cluster
```

Sortie :

```
{
    "serviceArns": [
        "arn:aws:ecs:region:aws_account_id:service/fargate-service"
```

```
    ]
}
```

Étape 6 : Décrire le service en cours d'exécution

Décrivez le service à l'aide du nom de service récupéré précédemment afin d'obtenir plus d'informations sur la tâche.

```
aws ecs describe-services --cluster fargate-cluster --services fargate-service
```

En cas de réussite, une description des défaillances de service et des services est renvoyée. Par exemple, dans la section des services, vous trouverez des informations sur les déploiements, telles que le statut des tâches en cours d'exécution ou en attente. Vous trouverez également des informations sur la définition des tâches, la configuration réseau et les événements horodatés. Dans la section des défaillances, vous trouverez des informations sur les défaillances, le cas échéant, associées à l'appel. Pour le dépannage, consultez Messages d'événements de service. Pour de plus amples informations sur la description du service, consultez Description des services.

```
{
    "services": [
        {
            "status": "ACTIVE",
            "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-definition/
sample-fargate-windows:1",
            "pendingCount": 2,
            "launchType": "FARGATE",
            "loadBalancers": [],
            "roleArn": "arn:aws:iam::aws_account_id:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
            "placementConstraints": [],
            "createdAt": 1510811361.128,
            "desiredCount": 2,
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-abcd1234"
                    ],
                    "securityGroups": [
                        "sg-abcd1234"
                    ],
                    "assignPublicIp": "DISABLED"
```

```
                }
            },
            "platformVersion": "LATEST",
            "serviceName": "fargate-service",
            "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster",
            "serviceArn": "arn:aws:ecs:region:aws_account_id:service/fargate-service",
            "deploymentConfiguration": {
                "maximumPercent": 200,
                "minimumHealthyPercent": 100
            },
            "deployments": [
                {
                    "status": "PRIMARY",
                    "networkConfiguration": {
                        "awsvpcConfiguration": {
                            "subnets": [
                                "subnet-abcd1234"
                            ],
                            "securityGroups": [
                                "sg-abcd1234"
                            ],
                            "assignPublicIp": "DISABLED"
                        }
                    },
                    "pendingCount": 2,
                    "launchType": "FARGATE",
                    "createdAt": 1510811361.128,
                    "desiredCount": 2,
                    "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-
definition/sample-fargate-windows:1",
                    "updatedAt": 1510811361.128,
                    "platformVersion": "0.0.1",
                    "id": "ecs-svc/9223370526043414679",
                    "runningCount": 0
                }
            ],
            "events": [
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 53c0de40-ea3b-489f-a352-623bf1235f08) (task d0aec985-901b-488f-9fb4-61b991b332a3).",
                    "id": "92b8443e-67fb-4886-880c-07e73383ea83",
                    "createdAt": 1510811841.408
                },
                {
```

```
                    "message": "(service fargate-service) has started 2 tasks: (task
 b4911bee-7203-4113-99d4-e89ba457c626) (task cc5853e3-6e2d-4678-8312-74f8a7d76474).",
                    "id": "d85c6ec6-a693-43b3-904a-a997e1fc844d",
                    "createdAt": 1510811601.938
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 cba86182-52bf-42d7-9df8-b744699e6cfc) (task f4c1ad74-a5c6-4620-90cf-2aff118df5fc).",
                    "id": "095703e1-0ca3-4379-a7c8-c0f1b8b95ace",
                    "createdAt": 1510811364.691
                }
            ],
            "runningCount": 0,
            "placementStrategy": []
        }
    ],
    "failures": []
}
```

Étape 7 : Nettoyer

Une fois que vous avez terminé ce didacticiel, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources inutilisées.

Supprimez le service.

```
aws ecs delete-service --cluster fargate-cluster --service fargate-service --force
```

Supprimez le cluster.

```
aws ecs delete-cluster --cluster fargate-cluster
```

Création d'une tâche Amazon ECS pour le type de EC2 lancement avec AWS CLI

Les étapes suivantes vous aideront à configurer un cluster, enregistrer une définition de tâche, exécuter une tâche et effectuer d'autres scénarios courants dans Amazon ECS avec la AWS CLI. Utilisez la dernière version du AWS CLI. Pour plus d'informations sur la mise à niveau vers la dernière version, voir Installation ou mise à jour vers la dernière version du AWS CLI.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir de AWS CLI SDKs, et de l'API Amazon ECS via les deux IPv4 et. IPv6 Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

Rubriques

- Prérequis
- Créer un cluster
- Lancer une instance de conteneur avec l'AMI Amazon ECS
- Lister les instances de conteneur
- Décrivez votre instance de conteneur
- Enregistrer une définition de tâche
- Lister les définitions de tâches
- Créer un service
- Lister les services
- Décrivez le service
- Décrire la tâche en cours
- Testez le serveur Web
- Nettoyage des ressources

Prérequis

Le didacticiel suppose de remplir les prérequis suivants :

- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations sur l'installation ou la mise à niveau de votre AWS CLI AWS CLI.
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique Amazon ECS_ FullAccess IAM.
- Un rôle IAM d'instance de conteneur a été créé pour être utilisé. Pour en savoir plus, consultezRôle IAM d'instance de conteneur Amazon ECS.

- Vous avez créé un VPC pour l'utiliser. Pour de plus amples informations, veuillez consulter the section called “Créer un Virtual Private Cloud”.
- (Facultatif) AWS CloudShell est un outil qui fournit aux clients une ligne de commande sans avoir à créer leur propre EC2 instance. Pour plus d'informations, voir Qu'est-ce que c'est AWS CloudShell ? dans le guide de AWS CloudShell l'utilisateur.

Créer un cluster

Par défaut, votre compte reçoit un cluster `default` lorsque vous lancez votre première instance de conteneur.

> **Note**
>
> L'utilisation du cluster `default` qui vous est fourni présente l'avantage que vous n'avez pas besoin de spécifier l'option `--cluster` *`cluster_name`* dans les commandes suivantes. Si vous créez votre propre cluster, sans valeur par défaut, vous devez spécifier `--cluster` *`cluster_name`* pour chaque commande que vous prévoyez d'utiliser avec ce cluster.

Créez votre propre cluster avec un nom unique à l'aide de la commande suivante :

```
aws ecs create-cluster --cluster-name MyCluster
```

Sortie :

```
{
    "cluster": {
        "clusterName": "MyCluster",
        "status": "ACTIVE",
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/MyCluster"
    }
}
```

Lancer une instance de conteneur avec l'AMI Amazon ECS

Les instances de conteneur sont EC2 des instances qui exécutent l'agent de conteneur Amazon ECS et qui ont été enregistrées dans un cluster. Dans cette section, vous allez lancer une EC2 instance à l'aide de l'AMI optimisée pour ECS.

Pour lancer une instance de conteneur avec AWS CLI

1. Récupérez le dernier ID AMI Amazon Linux 2 optimisé pour ECS à Région AWS l'aide de la commande suivante. Cette commande utilise le AWS Systems Manager Parameter Store pour obtenir le dernier ID d'AMI optimisé pour ECS. L'AMI inclut l'agent de conteneur Amazon ECS et le runtime Docker préinstallés.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
recommended --query 'Parameters[0].Value' --output text | jq -r '.image_id'
```

Sortie :

```
ami-abcd1234
```

2. Créez un groupe de sécurité qui autorise l'accès SSH pour gérer votre instance de conteneur et l'accès HTTP pour le serveur Web.

```
aws ec2 create-security-group --group-name ecs-tutorial-sg --description "ECS
 tutorial security group"
```

Sortie :

```
{
    "GroupId": "sg-abcd1234"
}
```

3. Ajoutez une règle entrante au groupe de sécurité en exécutant la commande suivante.

```
aws ec2 authorize-security-group-ingress --group-id sg-abcd1234 --protocol tcp --
port 80 --cidr 0.0.0.0/0
```

Sortie :

```
{
    "Return": true,
    "SecurityGroupRules": [
        {
            "SecurityGroupRuleId": "sgr-efgh5678",
            "GroupId": "sg-abcd1234",
            "GroupOwnerId": "123456789012",
```

```
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

Le groupe de sécurité autorise désormais l'accès SSH à partir de la plage d'adresses IP spécifiée et l'accès HTTP depuis n'importe où. Dans un environnement de production, vous devez restreindre l'accès SSH à votre adresse IP spécifique et envisager de limiter l'accès HTTP selon les besoins.

4. Créez une paire de EC2 clés pour accéder en SSH à votre instance de conteneur.

```
aws ec2 create-key-pair --key-name ecs-tutorial-key --query 'KeyMaterial' --output
 text > ecs-tutorial-key.pem
chmod 400 ecs-tutorial-key.pem
```

La clé privée est enregistrée sur votre machine locale avec les autorisations appropriées pour l'accès SSH.

5. Lancez une EC2 instance à l'aide de l'AMI optimisée pour ECS et configurez-la pour qu'elle rejoigne votre cluster.

```
aws ec2 run-instances --image-id ami-abcd1234 --instance-type t3.micro --key-
name ecs-tutorial-key --security-group-ids sg-abcd1234 --iam-instance-profile
 Name=ecsInstanceRole --user-data '#!/bin/bash
echo ECS_CLUSTER=MyCluster >> /etc/ecs/ecs.config'
{
    "Instances": [
        {
            "InstanceId": "i-abcd1234",
            "ImageId": "ami-abcd1234",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "PrivateDnsName": "",
            "PublicDnsName": "",
            "StateReason": {
```

```
                "Code": "pending",
                "Message": "pending"
            },
            "InstanceType": "t3.micro",
            "KeyName": "ecs-tutorial-key",
            "LaunchTime": "2025-01-13T10:30:00.000Z"
        }
    ]
}
```

Le script de données utilisateur configure l'agent Amazon ECS pour enregistrer l'instance auprès de votre`MyCluster`. L'instance utilise le rôle `ecsInstanceRole` IAM, qui fournit les autorisations nécessaires à l'agent.

Lister les instances de conteneur

Quelques minutes après le lancement de l'instance de conteneur, l'agent Amazon ECS enregistre l'instance avec votre cluster par défaut. Vous pouvez afficher les instances de conteneur d'un cluster en exécutant la commande suivante :

```
aws ecs list-container-instances --cluster default
```

Sortie :

```
{
    "containerInstanceArns": [
        "arn:aws:ecs:us-east-1:aws_account_id:container-instance/container_instance_ID"
    ]
}
```

Décrivez votre instance de conteneur

Après avoir obtenu l'ARN ou l'ID d'une instance de conteneur, vous pouvez utiliser la commande describe-container-instances pour obtenir des informations importantes sur l'instance, comme les ressources UC et de mémoire restantes et enregistrées.

```
aws ecs describe-container-instances --cluster MyCluster --container-
instances container_instance_ID
```

Sortie :

```
{
    "failures": [],
    "containerInstances": [
        {
            "status": "ACTIVE",
            "registeredResources": [
                {
                    "integerValue": 1024,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "CPU",
                    "doubleValue": 0.0
                },
                {
                    "integerValue": 995,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "MEMORY",
                    "doubleValue": 0.0
                },
                {
                    "name": "PORTS",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [
                        "22",
                        "2376",
                        "2375",
                        "51678"
                    ],
                    "type": "STRINGSET",
                    "integerValue": 0
                },
                {
                    "name": "PORTS_UDP",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [],
                    "type": "STRINGSET",
                    "integerValue": 0
                }
            ],
```

```
            "ec2InstanceId": "instance_id",
            "agentConnected": true,
            "containerInstanceArn": "arn:aws:ecs:us-west-2:aws_account_id:container-
instance/container_instance_ID",
            "pendingTasksCount": 0,
            "remainingResources": [
                {
                    "integerValue": 1024,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "CPU",
                    "doubleValue": 0.0
                },
                {
                    "integerValue": 995,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "MEMORY",
                    "doubleValue": 0.0
                },
                {
                    "name": "PORTS",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [
                        "22",
                        "2376",
                        "2375",
                        "51678"
                    ],
                    "type": "STRINGSET",
                    "integerValue": 0
                },
                {
                    "name": "PORTS_UDP",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [],
                    "type": "STRINGSET",
                    "integerValue": 0
                }
            ],
            "runningTasksCount": 0,
            "attributes": [
```

```
                {
                    "name": "com.amazonaws.ecs.capability.privileged-container"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
                },
                {
                    "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
                },
                {
                    "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
                }
            ],
            "versionInfo": {
                "agentVersion": "1.5.0",
                "agentHash": "b197edd",
                "dockerVersion": "DockerVersion: 1.7.1"
            }
        }
    ]
}
```

Vous pouvez également trouver l'ID d' EC2 instance Amazon que vous pouvez utiliser pour surveiller l'instance dans la EC2 console Amazon ou à l'aide de la aws ec2 describe-instances --instance-id ***instance_id*** commande.

Enregistrer une définition de tâche

Avant de pouvoir exécuter une tâche sur votre cluster Amazon ECS, vous devez enregistrer une définition de tâche. Les définitions de tâches sont des listes de conteneurs regroupés ensemble. L'exemple suivant est une définition de tâche simple qui utilise une `nginx` image. Pour plus d'informations sur les paramètres de définition des tâches disponibles, consultez Définitions de tâche Amazon ECS.

```
{
    "family": "nginx-task",
```

```
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/ecs-sample-image/amazon-ecs-sample:latest",
            "cpu": 256,
            "memory": 512,
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "requiresCompatibilities": ["EC2"],
    "networkMode": "bridge"
}
```

L'exemple JSON ci-dessus peut être transmis AWS CLI au de deux manières : vous pouvez enregistrer la définition de tâche JSON sous forme de fichier et la transmettre avec l'`--cli-input-json file://`*`path_to_file.json`*option. Vous pouvez également éviter les guillemets dans le JSON et transmettre les définitions des conteneurs JSON sur la ligne de commande. Si vous choisissez de transmettre les définitions de conteneur sur la ligne de commande, votre commande nécessite également un paramètre `--family` qui permet de conserver plusieurs versions de votre définition de tâche associées les unes aux autres.

Pour utiliser un fichier JSON pour les définitions de conteneur :

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/nginx.json
```

La commande register-task-definition renvoie une description de la définition de tâche une fois l'enregistrement terminé.

```
{
    "taskDefinition": {
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/nginx-
task:1",
        "family": "nginx-task",
        "revision": 1,
        "status": "ACTIVE",
```

```
        "containerDefinitions": [
            {
                "name": "nginx",
                "image": "public.ecr.aws/docker/library/nginx:latest",
                "cpu": 256,
                "memory": 512,
                "essential": true,
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80,
                        "protocol": "tcp"
                    }
                ],
                "environment": [],
                "mountPoints": [],
                "volumesFrom": []
            }
        ],
        "volumes": [],
        "networkMode": "bridge",
        "compatibilities": [
            "EC2"
        ],
        "requiresCompatibilities": [
            "EC2"
        ]
    }
}
```

Lister les définitions de tâches

Vous pouvez répertorier les définitions de tâches de votre compte à tout moment à l'aide de la commande list-task-definitions. Le résultat de cette commande indique les `revision` valeurs `family` et que vous pouvez utiliser ensemble lors de l'appelcreate-service.

```
aws ecs list-task-definitions
```

Sortie :

```
{
    "taskDefinitionArns": [
```

```
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep360:1",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep360:2",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/nginx-task:1",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:3",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:4",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:5",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:6"
    ]
}
```

Créer un service

Après avoir enregistré une tâche pour votre compte et lancé une instance de conteneur enregistrée dans votre cluster, vous pouvez créer un service Amazon ECS qui exécute et gère le nombre souhaité de tâches simultanément à l'aide de la définition de tâche que vous avez enregistrée. Pour cet exemple, vous placez une seule instance de la définition de tâche `nginx:1` dans votre cluster par défaut.

```
aws ecs create-service --cluster default --service-name nginx-service --task-
definition nginx-task:1 --desired-count 1
```

Sortie :

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:aws_account_id:service/MyCluster/nginx-
service",
        "serviceName": "nginx-service",
        "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/MyCluster",
        "taskDefinition": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/nginx-
task:1",
        "desiredCount": 1,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "EC2",
        "status": "ACTIVE",
        "createdAt": "2025-01-13T10:45:00.000Z"
    }
}
```

Lister les services

Affichez les services de votre cluster. Vous devriez voir le service que vous avez exécuté dans la section précédente. Vous pouvez noter l'ID de service ou l'ARN complet renvoyé par cette commande et l'utiliser pour décrire le service ultérieurement.

```
aws ecs list-services --cluster MyCluster
```

Sortie :

```
{
    "taskArns": [
        "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID"
    ]
}
```

Décrivez le service

Décrivez le service à l'aide de la commande suivante pour obtenir plus d'informations sur le service.

```
aws ecs describe-services --cluster MyCluster --services nginx-service
```

Sortie :

```
{
    "services": [
        {
            "serviceArn": "arn:aws:ecs:us-east-1:aws_account_id:service/MyCluster/
nginx-service",
            "serviceName": "nginx-service",
            "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/MyCluster",
            "taskDefinition": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/
nginx-task:1",
            "desiredCount": 1,
            "runningCount": 1,
            "pendingCount": 0,
            "launchType": "EC2",
            "status": "ACTIVE",
            "createdAt": "2025-01-13T10:45:00.000Z",
            "events": [
                {
```

```
                    "id": "abcd1234-5678-90ab-cdef-1234567890ab",
                    "createdAt": "2025-01-13T10:45:30.000Z",
                    "message": "(service nginx-service) has started 1 tasks: (task
 abcd1234-5678-90ab-cdef-1234567890ab)."
                }
            ]
        }
    ]
}
```

Décrire la tâche en cours

Après avoir décrit le service, exécutez la commande suivante pour obtenir plus d'informations sur la tâche exécutée dans le cadre de votre service.

```
aws ecs list-tasks --cluster MyCluster --service-name nginx-service
```

Sortie :

```
{
    "tasks": [
        {
            "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/MyCluster/
abcd1234-5678-90ab-cdef-1234567890ab",
            "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/MyCluster",
            "taskDefinitionArn": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/
nginx-task:1",
            "containerInstanceArn": "arn:aws:ecs:us-east-1:aws_account_id:container-
instance/MyCluster/abcd1234-5678-90ab-cdef-1234567890ab",
            "lastStatus": "RUNNING",
            "desiredStatus": "RUNNING",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-east-1:aws_account_id:container/
MyCluster/abcd1234-5678-90ab-cdef-1234567890ab/abcd1234-5678-90ab-cdef-1234567890ab",
                    "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/MyCluster/
abcd1234-5678-90ab-cdef-1234567890ab",
                    "name": "nginx",
                    "lastStatus": "RUNNING",
                    "networkBindings": [
                        {
                            "bindIP": "0.0.0.0",
```

```
                            "containerPort": 80,
                            "hostPort": 80,
                            "protocol": "tcp"
                        }
                    ]
                }
            ],
            "createdAt": "2025-01-13T10:45:00.000Z",
            "startedAt": "2025-01-13T10:45:30.000Z"
        }
    ]
}
```

Testez le serveur Web

Pour tester le serveur Web

1. Récupérez l'adresse IP publique de votre instance de conteneur en exécutant la commande suivante.

   ```
   aws ec2 describe-instances --instance-ids i-abcd1234 --query
    'Reservations[0].Instances[0].PublicIpAddress' --output text
   ```

   Sortie :

   ```
   203.0.113.25
   ```

2. Après avoir récupéré l'adresse IP, exécutez la `curl` commande suivante avec l'adresse IP.

   ```
   curl http://203.0.113.25
   ```

   Sortie :

   ```
   <!DOCTYPE html>
   <html>
   <head>
   <title>Welcome to nginx!</title>
   ...
   </head>
   <body>
   <h1>Welcome to nginx!</h1>
   ```

```
<p>If you can see this page, the nginx web server is successfully installed and
 working.</p>
...
</body>
</html>
```

La page d'accueil de nginx confirme que votre service fonctionne correctement et est accessible depuis Internet.

Nettoyage des ressources

Pour éviter d'encourir des frais, nettoyez les ressources que vous avez créées dans ce didacticiel.

Pour nettoyer des ressources

1. Mettez à jour le service pour qu'il n'y ait aucune tâche souhaitée, puis supprimez-le.

```
aws ecs update-service --cluster MyCluster --service nginx-service --desired-count
 0
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/MyCluster/nginx-
service",
        "serviceName": "nginx-service",
        "desiredCount": 0,
        "runningCount": 1,
        "pendingCount": 0,
        "status": "ACTIVE"
    }
}
```

2. Attendez que les tâches en cours d'exécution s'arrêtent, puis supprimez le service.

```
aws ecs delete-service --cluster MyCluster --service nginx-service
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/MyCluster/nginx-
service",
        "serviceName": "nginx-service",
        "status": "DRAINING"
    }
```

```
}
```

3. Mettez fin à l'instance de conteneur que vous avez créée.

```
aws ec2 terminate-instances --instance-ids i-abcd1234
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-abcd1234",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

4. Nettoyez le groupe de sécurité et la paire de clés que vous avez créés.

```
aws ec2 delete-security-group --group-id sg-abcd1234
aws ec2 delete-key-pair --key-name ecs-tutorial-key
rm ecs-tutorial-key.pem
```

5. Supprimez le cluster Amazon ECS.

```
aws ecs delete-cluster --cluster MyCluster
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
        "clusterName": "MyCluster",
        "status": "INACTIVE"
    }
}
```

Configuration d'Amazon ECS pour écouter CloudWatch les événements

Découvrez comment configurer une fonction Lambda simple qui écoute les événements liés aux tâches et les enregistre dans un flux de CloudWatch log Logs.

Prérequis : Configuration d'un cluster test

Si vous ne possédez pas de cluster en cours d'exécution à partir duquel capturer des événements, suivez les étapes dans the section called “Création d'un cluster pour le type de lancement Fargate” pour en créer un. À la fin de ce didacticiel, vous exécutez une tâche sur ce cluster pour tester que la configuration de votre fonction Lambda est correcte.

Étape 1 : Créer la fonction Lambda

Au cours de cette procédure, vous allez créer une fonction Lambda simple à utiliser comme cible pour les messages de flux d'événements Amazon ECS.

1. Ouvrez la AWS Lambda console à l'adresse https://console.aws.amazon.com/lambda/.
2. Sélectionnez Create function (Créer une fonction).
3. Dans l'écran Author from scratch (Créer à partir de zéro), procédez comme suit :
   a. Pour Name (Nom), saisissez une valeur.
   b. Pour Runtime (Exécution), choisissez votre version de Python, par exemple, Python 3.9.
   c. Pour Role (Rôle), choisissez Create a new role with basic Lambda permissions (Créer un nouveau rôle avec les autorisations Lambda de base).
4. Sélectionnez Create function (Créer une fonction).
5. Dans la section Code de fonction, modifiez l'exemple de code selon l'exemple suivant :

```
import json

def lambda_handler(event, context):
    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source
 type of: aws.ecs")

    print('Here is the event:')
```

```
    print(json.dumps(event))
```

Il s'agit d'une fonction Python 3.9 simple qui imprime les événements envoyés par Amazon ECS. Si tout est correctement configuré, à la fin de ce didacticiel, vous verrez que les détails de l'événement apparaissent dans le flux de log CloudWatch Logs associé à cette fonction Lambda.

6. Choisissez Enregistrer.

Étape 2 : Enregistrer une règle d'événement

Ensuite, vous créez une règle d' CloudWatch événements qui capture les événements de tâches provenant de vos clusters Amazon ECS. Cette règle capture tous les événements provenant de tous les clusters du compte dans lequel il est défini. Les messages de tâche eux-mêmes contiennent des informations sur la source de l'événement, y compris le cluster sur lequel elle réside, que vous pouvez utiliser pour filtrer et trier les événements par programmation.

> **Note**
>
> Lorsque vous utilisez la règle AWS Management Console pour créer un événement, la console ajoute automatiquement les autorisations IAM nécessaires pour autoriser CloudWatch Events à appeler votre fonction Lambda. Si vous créez une règle d'événement à l'aide du AWS CLI, vous devez accorder cette autorisation de manière explicite. Pour plus d'informations, consultez les sections Événements sur Amazon EventBridge et Modèles d' EventBridge événements Amazon dans le guide de EventBridge l'utilisateur Amazon.

Pour acheminer des événements vers votre fonction Lambda

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, choisissez Events (Événements), Rules (Règles), Create rule (Créer une règle).
3. Pour Event Source (Source de l'événement), choisissez ECS comme source d'événement. Par défaut, la règle s'applique à tous les événements Amazon ECS pour tous vos groupes Amazon ECS. Sinon, vous pouvez sélectionner des événements spécifiques ou un groupe Amazon ECS spécifique.
4. Pour Targets (Cibles), choisissez Add target (Ajouter une cible), pour Target type (Type de cible), choisissez Lambda function (Fonction Lambda), puis sélectionnez votre fonction Lambda.

5. Choisissez Configure details (Configurer les détails).
6. Pour Rule definition (Définition de règle), saisissez un nom et une description pour la règle, puis choisissez Create rule (Créer une règle).

Étape 3 : Créer une définition de tâche

Créez une définition de tâche.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, sélectionnez Task Definitions (Définition des tâches).
3. Choisissez Create new Task Definition (Créer une nouvelle définition de tâche), puis Create new revision with JSON (Créer une nouvelle révision avec JSON).
4. Copiez et collez l'exemple de définition de tâche suivant dans la zone, puis choisissez Save (Enregistrer).

```
{
   "containerDefinitions": [
      {
         "command": [
            "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample App</
title> <style>body {margin-top: 40px; background-color: #333;} </style> </
head><body> <div style=color:white;text-align:center> <h1>Amazon ECS Sample App</
h1> <h2>Congratulations!</h2> <p>Your application is now running on a container in
 Amazon ECS.</p> </div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
         ],
         "entryPoint": [
            "sh",
            "-c"
         ],
         "essential": true,
         "image": "public.ecr.aws/docker/library/httpd:2.4",
         "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
               "awslogs-group" : "/ecs/fargate-task-definition",
               "awslogs-region": "us-east-1",
               "awslogs-stream-prefix": "ecs"
            }
         },
```

```
            "name": "sample-fargate-app",
            "portMappings": [
               {
                  "containerPort": 80,
                  "hostPort": 80,
                  "protocol": "tcp"
               }
            ]
         }
      ],
      "cpu": "256",
      "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
      "family": "fargate-task-definition",
      "memory": "512",
      "networkMode": "awsvpc",
      "runtimePlatform": {
           "operatingSystemFamily": "LINUX"
       },
      "requiresCompatibilities": [
          "FARGATE"
       ]
   }
```

5. Choisissez Créer.

Étape 4 : Test de la règle

Enfin, vous créez une règle d' CloudWatch événements qui capture les événements de tâches provenant de vos clusters Amazon ECS. Cette règle capture tous les événements provenant de tous les clusters du compte dans lequel il est défini. Les messages de tâche eux-mêmes contiennent des informations sur la source de l'événement, y compris le cluster sur lequel elle réside, que vous pouvez utiliser pour filtrer et trier les événements par programmation.

Pour tester la règle

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Choisissez Task definitions (Définitions des tâches).
3. Choisissez console-sample-app-static, puis choisissez Déployer, Exécuter une nouvelle tâche.
4. Pour Cluster, choisissez par défaut, puis choisissez Deploy (Déployer).
5. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.

6. Dans le volet de navigation, choisissez Logs et sélectionnez le groupe de journaux pour votre fonction Lambda (par exemple, ***my-function***/aws/lambda/).
7. Sélectionnez un flux de journaux pour afficher les données d'événement.

Envoi d'alertes Amazon Simple Notification Service pour les événements liés à l'arrêt d'une tâche Amazon ECS

Configurez une règle d' EventBridge événement Amazon qui capture uniquement les événements de tâche lorsque la tâche a cessé de s'exécuter parce que l'un de ses conteneurs essentiels s'est arrêté. L'événement envoie uniquement les événements de tâche ayant une propriété spécifique `stoppedReason` à la rubrique Amazon SNS désignée.

Prérequis : Configuration d'un cluster test

Si vous ne possédez pas de cluster en cours d'exécution à partir duquel capturer des événements, suivez les étapes dans Mise en route avec la console à l'aide de conteneurs Linux sur AWS Fargate pour en créer un. À la fin de ce didacticiel, vous exécuterez une tâche sur ce cluster pour vérifier que vous avez correctement configuré votre rubrique et votre EventBridge règle Amazon SNS.

Prérequis : configurer les autorisations pour Amazon SNS

EventBridge Pour autoriser la publication sur une rubrique Amazon SNS, utilisez les commandes aws sns get-topic-attributes et aws sns. set-topic-attributes

Pour obtenir des informations sur l’ajout de l’autorisation, consultez Autorisations Amazon SNS dans le Guide du développeur Amazon Simple Notification Service.

Ajoutez les autorisations suivantes :

```
{
  "Sid": "PublishEventsToMyTopic",
  "Effect": "Allow",
  "Principal": {
     "Service": "events.amazonaws.com"
  },
  "Action": "sns: Publish",
  "Resource": "arn:aws:sns:region:account-id:TaskStoppedAlert",
}
```

Étape 1 : Créer une rubrique Amazon SNS et s'y abonner

Pour ce didacticiel, vous configurez une rubrique Amazon SNS à utiliser comme une cible de l'événement pour votre nouvelle règle d'événement.

Pour plus d'informations sur la façon de créer et de s'abonner à une rubrique Amazon SNS, consultez Démarrer avec Amazon SNS dans le Guide du développeur Amazon Simple Notification Service et utilisez le tableau suivant pour déterminer les options à sélectionner.

Option	Valeur
Type	Standard
Nom	TaskStoppedAlert
Protocole	E-mails
Point de terminaison	Une adresse e-mail à laquelle vous avez actuellement accès

Étape 2 : Enregistrer une règle d'événement

Ensuite, vous enregistrez une règle d'événement qui capture uniquement les événements d'arrêt de la tâche pour les tâches avec des conteneurs arrêtés.

Pour plus d'informations sur la création et l'abonnement à une rubrique Amazon SNS, consultez la section Créer une règle EventBridge dans Amazon dans le guide de EventBridge l'utilisateur Amazon et utilisez le tableau suivant pour déterminer les options à sélectionner.

Option	Valeur
Type de règle	Règle avec un modèle d'événement
Source de l’événement	AWS événements ou événements EventBridge partenaires

Option	Valeur
Modèle d'événement	Modèle personnalisé (éditeur JSON)
Modèle d'événement	`{ "source":[ "aws.ecs" ], "detail-type":[ "ECS Task State Change" ], "detail":{ "lastStatus":[ "STOPPED" ], "stoppedReason":[ "Essential container in task exited" ] } }`
Type de cible	AWS service
Cible	Rubrique SNS
Rubrique	TaskStoppedAlert (Le sujet que vous avez créé à l'étape 1)

Étape 3 : Test de la règle

Vérifiez que la règle fonctionne en exécutant une tâche qui se ferme peu après son démarrage. Si votre règle d'événement est configurée correctement, vous recevez un message électronique en quelques minutes avec le texte de l'événement. Si vous disposez d'une définition de tâche existante qui peut satisfaire aux exigences de la règle, exécutez une tâche à l'aide de celle-ci. Si vous ne le

faites pas, les étapes suivantes vous permettront d'enregistrer une définition de tâche Fargate et de l'exécuter.

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
4. Dans la zone de l'éditeur JSON, modifiez votre fichier JSON, copiez ce qui suit dans l'éditeur.

```
{
   "containerDefinitions":[
      {
         "command":[
            "sh",
            "-c",
            "sleep 5"
         ],
         "essential":true,
         "image":"public.ecr.aws/amazonlinux/amazonlinux:latest",
         "name":"test-sleep"
      }
   ],
   "cpu":"256",
   "executionRoleArn":"arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
   "family":"fargate-task-definition",
   "memory":"512",
   "networkMode":"awsvpc",
   "requiresCompatibilities":[
      "FARGATE"
   ]
}
```

5. Choisissez Créer.

Pour exécuter une tâche à partir de la console

1. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
2. Sur la page Clusters, choisissez le cluster que vous avez créé dans les conditions préalables.
3. Sous l'onglet Tasks (Tâches), choisissez Run new task (Exécuter une nouvelle tâche).
4. Pour Application type (Type d'application), choisissez Task (Tâche).

5. Pour Définition de la tâche, sélectionnez fargate-task-definition.
6. Pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer.
7. Choisissez Créer.

Concaténation de messages de journal Amazon ECS multilignes ou empilés

À partir de AWS la version 2.22.0 de Fluent Bit, un filtre multiligne est inclus. Le filtre multiligne aide à concaténer les messages de journaux qui appartiennent à l'origine à un seul contexte, mais qui ont été divisés en plusieurs enregistrements ou lignes de journal. Pour plus d'informations sur le filtre multiligne, consultez la documentation de Fluent Bit.

Voici des exemples courants de messages de journaux divisés :

- Suivis de pile.
- Applications qui impriment des journaux sur plusieurs lignes.
- Messages de journal qui ont été divisés, car ils étaient plus longs que la taille maximale de mémoire tampon d'exécution spécifiée. Vous pouvez concaténer les messages de journal divisés par le moteur d'exécution du conteneur en suivant l'exemple suivant GitHub : FireLens Exemple : Partial/Split Concaténer les journaux de conteneur.

Autorisations IAM requises

Vous disposez des autorisations IAM nécessaires pour que l'agent du conteneur extrait les images du conteneur depuis Amazon ECR et pour que le conteneur achemine les journaux vers CloudWatch Logs.

Pour ces autorisations, vous devez disposer des rôles suivants :

- Un rôle IAM de tâche.
- Rôle IAM d'exécution de tâches.

Pour utiliser l’éditeur de politique JSON afin de créer une politique

1. Connectez-vous à la console IAM AWS Management Console et ouvrez-la à https://console.aws.amazon.com/iam/l'adresse.

2. Dans le panneau de navigation de gauche, sélectionnez Policies (Politiques).

   Si vous sélectionnez Politiques pour la première fois, la page Bienvenue dans les politiques gérées s'affiche. Sélectionnez Mise en route.

3. En haut de la page, sélectionnez Créer une politique.
4. Dans la section Éditeur de politique, choisissez l'option JSON.
5. Entrez le document de politique JSON suivant :

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": [
   "logs:CreateLogStream",
   "logs:CreateLogGroup",
   "logs:PutLogEvents"
  ],
  "Resource": "*"
 }]
}
```

6. Choisissez Suivant.

   > **Note**
   >
   > Vous pouvez basculer à tout moment entre les options des éditeurs visuel et JSON. Toutefois, si vous apportez des modifications ou si vous choisissez Suivant dans l'éditeur visuel, IAM peut restructurer votre politique afin de l'optimiser pour l'éditeur visuel. Pour plus d'informations, consultez la page Restructuration de politique dans le Guide de l'utilisateur IAM.

7. Sur la page Vérifier et créer, saisissez un Nom de politique et une Description (facultative) pour la politique que vous créez. Vérifiez les Autorisations définies dans cette politique pour voir les autorisations accordées par votre politique.
8. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Déterminez à quel moment utiliser le paramètre de journal multiligne

Vous trouverez ci-dessous des exemples d'extraits de journal que vous pouvez voir dans la console CloudWatch Logs avec le paramètre de journal par défaut. Vous pouvez regarder la ligne qui commence par `log` pour déterminer si vous avez besoin du filtre multiligne. Lorsque le contexte est le même, vous pouvez utiliser le paramètre de journal multiligne. Dans cet exemple, le contexte est « com.myproject.model ». MyProject».

```
2022-09-20T15:47:56:595-05-00                           {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-
app", "source=": "stdout", "log": ": "     at com.myproject.modele.
(MyProject.badMethod.java:22)",
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log": ": "     at com.myproject.model.MyProject.badMethod(MyProject.java:22)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE",
      "ecs_task_definition": "firelense-example-multiline:3"
     }
```

```
2022-09-20T15:47:56:595-05-00                           {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-app", "stdout",
 "log": ": "     at com.myproject.modele.(MyProject.oneMoreMethod.java:18)",
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log": ": "     at
 com.myproject.model.MyProject.oneMoreMethod(MyProject.java:18)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE,
      "ecs_task_definition": "firelense-example-multiline:3"
     }
```

Une fois que vous avez utilisé le paramètre de journal multiligne, la sortie ressemblera à l'exemple ci-dessous.

```
2022-09-20T15:47:56:595-05-00                           {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-app",
 "stdout",...
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log:    "September 20, 2022 06:41:48 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!\n
    at com.myproject.module.MyProject.badMethod(MyProject.java:22)\n    at
    at com.myproject.model.MyProject.oneMoreMethod(MyProject.java:18)
 com.myproject.module.MyProject.main(MyProject.java:6)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE",
      "ecs_task_definition": "firelense-example-multiline:2"
     }
```

Options d'analyse et de concaténation

Pour analyser les journaux et concaténer des lignes qui ont été divisées en raison de sauts de lignes, vous pouvez utiliser l'une de ces deux options.

- Utiliser votre propre fichier d'analyseur qui contient les règles pour analyser et concaténer les lignes qui appartiennent au même message.
- Utiliser un analyseur intégré de Fluent Bit. Pour une liste des langues supportées par les analyseurs intégrés de Fluent Bit, consultez la documentation de Fluent Bit.

Le didacticiel suivant vous guide à travers les étapes pour chaque cas d'utilisation. Les étapes vous montrent comment concaténer des lignes multiples et envoyer les journaux à Amazon. CloudWatch Vous pouvez spécifier une destination différente pour vos journaux.

Exemple : Utiliser un analyseur que vous créez

Dans cet exemple, vous allez réaliser les étapes suivantes :

1. Créer et charger l'image d'un conteneur Fluent Bit.

2. Créer et charger l'image d'une application multiligne de démonstration qui s'exécute, échoue, et génère un suivi de pile multiligne.
3. Créer la définition de tâche et exécuter la tâche.
4. Afficher les journaux pour vérifier que les messages qui couvrent plusieurs lignes apparaissent concaténés.

Création et chargement de l'image d'un conteneur Fluent Bit

Cette image inclura le fichier d'analyseur où vous spécifiez l'expression régulière et un fichier de configuration qui fait référence au fichier d'analyseur.

1. Créez un dossier avec le nom `FluentBitDockerImage`.
2. Dans ce dossier, créez un fichier d'analyseur qui contient les règles pour analyser le journal et concaténer les lignes qui appartiennent au même message.

   a. Collez le contenu suivant dans le fichier d'analyseur :

```
[MULTILINE_PARSER]
    name          multiline-regex-test
    type          regex
    flush_timeout 1000
    #
    # Regex rules for multiline parsing
    # ---------------------------------
    #
    # configuration hints:
    #
    #  - first state always has the name: start_state
    #  - every field in the rule must be inside double quotes
    #
    # rules |   state name  | regex pattern                  | next state
    # ------|---------------|--------------------------------------------
    rule      "start_state"   "/(Dec \d+ \d+\:\d+\:\d+)(.*)/"  "cont"
    rule      "cont"          "/^\s+at.*/"                     "cont"
```

   Lorsque vous personnalisez votre modèle d'expression régulière, nous vous recommandons d'utiliser un éditeur d'expressions régulières pour tester l'expression.

   b. Enregistrez le fichier sous le nom `parsers_multiline.conf`.

3. Dans le dossier `FluentBitDockerImage`, créez un fichier de configuration personnalisé qui fait référence au fichier d'analyseur que vous avez créé à l'étape précédente.

   Pour plus d'informations sur le fichier de configuration personnalisé, consultez Spécification d'un fichier de configuration personnalisé dans le Guide du développeur Amazon Elastic Container Service

   a. Collez le contenu suivant dans le fichier :

```
[SERVICE]
    flush                 1
    log_level             info
    parsers_file          /parsers_multiline.conf

[FILTER]
    name                  multiline
    match                 *
    multiline.key_content log
    multiline.parser      multiline-regex-test
```

> **Note**
>
> Vous devez utiliser le chemin absolu de l'analyseur.

   b. Enregistrez le fichier sous le nom `extra.conf`.

4. Dans le dossier `FluentBitDockerImage`, créez le Dockerfile avec l'image Fluent Bit, l'analyseur et les fichiers de configuration que vous avez créés.

   a. Collez le contenu suivant dans le fichier :

```
FROM public.ecr.aws/aws-observability/aws-for-fluent-bit:latest

ADD parsers_multiline.conf /parsers_multiline.conf
ADD extra.conf /extra.conf
```

   b. Enregistrez le fichier sous le nom `Dockerfile`.

5. En utilisant le Dockerfile, créez une image Fluent Bit personnalisée avec l'analyseur et les fichiers de configuration personnalisés inclus.

> **Note**
>
> Vous pouvez placer le fichier d'analyseur et le fichier de configuration n'importe où dans l'image Docker, sauf lorsque `/fluent-bit/etc/fluent-bit.conf` ce chemin de fichier est utilisé par. FireLens

a. Créez l'image : `docker build -t fluent-bit-multiline-image.`

   Où : `fluent-bit-multiline-image` est le nom de l'image dans cet exemple.

b. Vérifiez que l'image a été créée correctement : `docker images –filter reference=fluent-bit-multiline-image`

   En cas de succès, la sortie montre l'image et l'identification `latest`.

6. Chargez l'image personnalisée Fluent Bit dans Amazon Elastic Container Registry.

   a. Créez un référentiel Amazon ECR pour stocker l'image : `aws ecr create-repository --repository-name fluent-bit-multiline-repo --region us-east-1`

      Où : `fluent-bit-multiline-repo` est le nom du référentiel et `us-east-1` est la région dans cet exemple.

      La sortie vous donne les détails du nouveau référentiel.

   b. Étiquetez votre image avec la valeur `repositoryUri` de la sortie précédente : `docker tag` *`fluent-bit-multiline-image repositoryUri`*

      Exemple : `docker tag fluent-bit-multiline-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

   c. Exécutez l'image docker pour vérifier qu'elle s'est exécutée correctement :`docker images –filter reference=`*`repositoryUri`*

      Dans le résultat, le nom du référentiel passe de fluent-bit-multiline-repo à`repositoryUri`.

   d. Authentifiez-vous auprès d'Amazon ECR en exécutant la commande `aws ecr get-login-password` et en spécifiant l'ID de registre auquel vous voulez vous authentifier : `aws ecr get-login-password | docker login --username AWS --password-stdin` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com`

Exemple : `ecr get-login-password | docker login --username AWS --password-stdin xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com`

Un message de connexion réussie apparaît.

e. Envoyez (push) l'image vers Amazon ECR : `docker push registry ID.dkr.ecr.region.amazonaws.com/repository name`

Exemple : `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

Création et chargement de l'image pour une application multiligne de démonstration

Cette image comprendra un fichier script Python qui exécute l'application et un exemple de fichier journal.

Lorsque vous exécutez la tâche, l'application simule l'exécution, puis échoue et crée un suivi de pile.

1. Créez un dossier nommé `multiline-app` : `mkdir multiline-app`
2. Créez un fichier script Python.

   a. Dans le dossier `multiline-app`, créez un fichier et nommez-le `main.py`.

   b. Collez le contenu suivant dans le fichier :

```
import os
import time
file1 = open('/test.log', 'r')
Lines = file1.readlines()

count = 0

for i in range(10):
    print("app running normally...")
    time.sleep(1)

# Strips the newline character
for line in Lines:
    count += 1
    print(line.rstrip())
print(count)
print("app terminated.")
```

c. Enregistrez le fichier `main.py`.

3. Créez un fichier journal d'exemple.

   a. Dans le dossier `multiline-app`, créez un fichier et nommez-le `test.log`.

   b. Collez le contenu suivant dans le fichier :

```
single line...
Dec 14 06:41:08 Exception in thread "main" java.lang.RuntimeException:
 Something has gone wrong, aborting!
    at com.myproject.module.MyProject.badMethod(MyProject.java:22)
    at com.myproject.module.MyProject.oneMoreMethod(MyProject.java:18)
    at com.myproject.module.MyProject.anotherMethod(MyProject.java:14)
    at com.myproject.module.MyProject.someMethod(MyProject.java:10)
    at com.myproject.module.MyProject.main(MyProject.java:6)
another line...
```

   c. Enregistrez le fichier `test.log`.

4. Dans le dossier `multiline-app`, créez le Dockerfile.

   a. Collez le contenu suivant dans le fichier :

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
ADD test.log /test.log

RUN yum upgrade -y && yum install -y python3

WORKDIR /usr/local/bin

COPY main.py .

CMD ["python3", "main.py"]
```

   b. Enregistrez le fichier `Dockerfile`.

5. À l'aide du Dockerfile, créez une image.

   a. Créez l'image : `docker build -t multiline-app-image`

      Où : `multiline-app-image` est le nom de l'image dans cet exemple.

b. Vérifiez que l'image a été créée correctement : `docker images –filter reference=multiline-app-image`

   En cas de succès, la sortie montre l'image et l'identification `latest`.

6. Chargez l'image dans Amazon Elastic Container Registry.

   a. Créez un référentiel Amazon ECR pour stocker l'image : `aws ecr create-repository --repository-name multiline-app-repo --region us-east-1`

      Où : `multiline-app-repo` est le nom du référentiel et `us-east-1` est la région dans cet exemple.

      La sortie vous donne les détails du nouveau référentiel. Notez la valeur de `repositoryUri`, car vous en aurez besoin dans les étapes suivantes.

   b. Étiquetez votre image avec la valeur `repositoryUri` de la sortie précédente : `docker tag` *`multiline-app-image repositoryUri`*

      Exemple : `docker tag multiline-app-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

   c. Exécutez l'image docker pour vérifier qu'elle s'est exécutée correctement :`docker images –filter reference=`*`repositoryUri`*

      Dans la sortie, le nom du référentiel passe de `multiline-app-repo` à la valeur de `repositoryUri`.

   d. Envoyez (push) l'image vers Amazon ECR : `docker push` *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

      Exemple : `docker push` *`xxxxxxxxxxxx`*`.dkr.ecr.`*`us-east-1`*`.amazonaws.com/multiline-app-repo`

Création de la définition de tâche et exécution de la tâche

1. Créez un fichier de définition de tâche avec le nom de fichier `multiline-task-definition.json`.
2. Collez le contenu suivant dans le fichier `multiline-task-definition.json` :

```
{
    "family": "firelens-example-multiline",
```

```
    "taskRoleArn": "task role ARN,
    "executionRoleArn": "execution role ARN",
    "containerDefinitions": [
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-
multiline-image:latest",
            "name": "log_router",
            "firelensConfiguration": {
                "type": "fluentbit",
                "options": {
                    "config-file-type": "file",
                    "config-file-value": "/extra.conf"
                }
            },
            "memoryReservation": 50
        },
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/multiline-app-
image:latest",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "cloudwatch_logs",
                    "region": "us-east-1",
                    "log_group_name": "multiline-test/application",
                    "auto_create_group": "true",
                    "log_stream_prefix": "multiline-"
                }
            },
            "memoryReservation": 100
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512"
}
```

Remplacez les éléments suivants dans la définition de tâche `multiline-task-definition.json` :

a. *task role ARN*

   Pour trouver l'ARN du rôle de la tâche, allez dans la console IAM. Choisissez Roles (Rôles) et trouvez le rôle de tâche `ecs-task-role-for-firelens` que vous avez créé. Choisissez le rôle et copiez l'ARN qui apparaît dans la section Summary (Résumé).

b. *execution role ARN*

   Pour trouver l'ARN du rôle d'exécution, allez dans la console IAM. Choisissez Roles (Rôles) et trouvez le rôle `ecsTaskExecutionRole`. Choisissez le rôle et copiez l'ARN qui apparaît dans la section Summary (Résumé).

c. *aws_account_id*

   Pour trouver votre `aws_account_id`, connectez-vous à la AWS Management Console. Choisissez votre nom d'utilisateur en haut à droite et copiez votre ID de compte.

d. *us-east-1*

   Remplacez la région si nécessaire.

3. Enregistrez le fichier de définition de tâche : `aws ecs register-task-definition --cli-input-json file://multiline-task-definition.json --region` *region*
4. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
5. Dans le panneau de navigation, choisissez Task Definitions (Définitions de tâches), puis la famille `firelens-example-multiline`, car nous avons enregistré la définition de tâche à cette famille dans la première ligne de la définition de tâche ci-dessus.
6. Choisissez la dernière version.
7. Choisissez Déployer, Exécuter la tâche.
8. Sur la page Exécuter la tâche, pour Cluster, choisissez le cluster, puis sous Mise en réseau, pour Sous-réseaux, choisissez les sous-réseaux disponibles pour votre tâche.
9. Choisissez Créer.

Vérifiez que les messages de journal multilignes dans Amazon CloudWatch apparaissent concaténés

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, développez Logs (Journaux) et choisissez Log groups (Groupes de journaux).
3. Choisissez le groupe de journaux `multiline-test/applicatio`.

4. Choisissez le journal. Affichez les messages. Les lignes qui correspondent aux règles du fichier analyseur sont concaténées et apparaissent comme un seul message.

   L'extrait de journal suivant montre les lignes concaténées dans un seul événement de suivi de pile Java :

```
{
    "container_id": "xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "log": "Dec 14 06:41:08 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!\n
 at com.myproject.module.MyProject.badMethod(MyProject.java:22)\n    at
 com.myproject.module.MyProject.oneMoreMethod(MyProject.java:18)\n
 at com.myproject.module.MyProject.anotherMethod(MyProject.java:14)\n
 at com.myproject.module.MyProject.someMethod(MyProject.java:10)\n    at
 com.myproject.module.MyProject.main(MyProject.java:6)",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:2"
}
```

   L'extrait de journal suivant montre comment le même message apparaît avec une seule ligne si vous exécutez un conteneur Amazon ECS qui n'est pas configuré pour concaténer les messages de journaux multilignes.

```
{
    "log": "Dec 14 06:41:08 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!",
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:3"
}
```

Exemple : Utilisation d'un analyseur intégré de Fluent Bit

Dans cet exemple, vous allez réaliser les étapes suivantes :

1. Créer et charger l'image d'un conteneur Fluent Bit.
2. Créer et charger l'image d'une application multiligne de démonstration qui s'exécute, échoue, et génère un suivi de pile multiligne.
3. Créer la définition de tâche et exécuter la tâche.
4. Afficher les journaux pour vérifier que les messages qui couvrent plusieurs lignes apparaissent concaténés.

Création et chargement de l'image d'un conteneur Fluent Bit

Cette image comprendra un fichier de configuration qui fait référence à l'analyseur de Fluent Bit.

1. Créez un dossier avec le nom `FluentBitDockerImage`.
2. Dans le dossier `FluentBitDockerImage`, créez un fichier de configuration personnalisé qui fait référence au fichier d'analyseur intégré de Fluent Bit.

   Pour plus d'informations sur le fichier de configuration personnalisé, consultez Spécification d'un fichier de configuration personnalisé dans le Guide du développeur Amazon Elastic Container Service

   a. Collez le contenu suivant dans le fichier :

   ```
   [FILTER]
       name                  multiline
       match                 *
       multiline.key_content log
       multiline.parser      go
   ```

   b. Enregistrez le fichier sous le nom `extra.conf`.
3. Dans le dossier `FluentBitDockerImage`, créez le Dockerfile avec l'image Fluent Bit, l'analyseur et les fichiers de configuration que vous avez créés.

   a. Collez le contenu suivant dans le fichier :

   ```
   FROM public.ecr.aws/aws-observability/aws-for-fluent-bit:latest
   ADD extra.conf /extra.conf
   ```

   b. Enregistrez le fichier sous le nom `Dockerfile`.
4. En utilisant le Dockerfile, créez une image Fluent Bit personnalisée avec le fichier de configuration personnalisé inclus.

> **Note**
>
> Vous pouvez placer le fichier de configuration n'importe où dans l'image Docker, sauf si `/fluent-bit/etc/fluent-bit.conf` ce chemin de fichier est utilisé par FireLens.

a. Créez l'image : `docker build -t fluent-bit-multiline-image.`

   Où : `fluent-bit-multiline-image` est le nom de l'image dans cet exemple.

b. Vérifiez que l'image a été créée correctement : `docker images –filter reference=fluent-bit-multiline-image`

   En cas de succès, la sortie montre l'image et l'identification `latest`.

5. Chargez l'image personnalisée Fluent Bit dans Amazon Elastic Container Registry.

   a. Créez un référentiel Amazon ECR pour stocker l'image : `aws ecr create-repository --repository-name fluent-bit-multiline-repo --region us-east-1`

      Où : `fluent-bit-multiline-repo` est le nom du référentiel et `us-east-1` est la région dans cet exemple.

      La sortie vous donne les détails du nouveau référentiel.

   b. Étiquetez votre image avec la valeur `repositoryUri` de la sortie précédente : `docker tag` *`fluent-bit-multiline-image repositoryUri`*

      Exemple : `docker tag fluent-bit-multiline-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

   c. Exécutez l'image docker pour vérifier qu'elle s'est exécutée correctement :`docker images –filter reference=`*`repositoryUri`*

      Dans le résultat, le nom du référentiel passe de fluent-bit-multiline-repo à`repositoryUri`.

   d. Authentifiez-vous auprès d'Amazon ECR en exécutant la commande `aws ecr get-login-password` et en spécifiant l'ID de registre auquel vous voulez vous authentifier : `aws ecr get-login-password | docker login --username AWS --password-stdin` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com`

Exemple : `ecr get-login-password | docker login --username AWS --password-stdin xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com`

Un message de connexion réussie apparaît.

e. Envoyez (push) l'image vers Amazon ECR : `docker push registry ID.dkr.ecr.region.amazonaws.com/repository name`

Exemple : `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

Création et chargement de l'image pour une application multiligne de démonstration

Cette image comprendra un fichier script Python qui exécute l'application et un exemple de fichier journal.

1. Créez un dossier nommé `multiline-app` : `mkdir multiline-app`
2. Créez un fichier script Python.

   a. Dans le dossier `multiline-app`, créez un fichier et nommez-le `main.py`.

   b. Collez le contenu suivant dans le fichier :

```
import os
import time
file1 = open('/test.log', 'r')
Lines = file1.readlines()

count = 0

for i in range(10):
    print("app running normally...")
    time.sleep(1)

# Strips the newline character
for line in Lines:
    count += 1
    print(line.rstrip())
print(count)
print("app terminated.")
```

c. Enregistrez le fichier `main.py`.

3. Créez un fichier journal d'exemple.

   a. Dans le dossier `multiline-app`, créez un fichier et nommez-le `test.log`.

   b. Collez le contenu suivant dans le fichier :

```
panic: my panic

goroutine 4 [running]:
panic(0x45cb40, 0x47ad70)
  /usr/local/go/src/runtime/panic.go:542 +0x46c fp=0xc42003f7b8 sp=0xc42003f710
 pc=0x422f7c
main.main.func1(0xc420024120)
  foo.go:6 +0x39 fp=0xc42003f7d8 sp=0xc42003f7b8 pc=0x451339
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003f7e0
 sp=0xc42003f7d8 pc=0x44b4d1
created by main.main
  foo.go:5 +0x58

goroutine 1 [chan receive]:
runtime.gopark(0x4739b8, 0xc420024178, 0x46fcd7, 0xc, 0xc420028e17, 0x3)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc420053e30 sp=0xc420053e00
 pc=0x42503c
runtime.goparkunlock(0xc420024178, 0x46fcd7, 0xc, 0x1000f010040c217, 0x3)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc420053e70 sp=0xc420053e30
 pc=0x42512e
runtime.chanrecv(0xc420024120, 0x0, 0xc420053f01, 0x4512d8)
  /usr/local/go/src/runtime/chan.go:506 +0x304 fp=0xc420053f20 sp=0xc420053e70
 pc=0x4046b4
runtime.chanrecv1(0xc420024120, 0x0)
  /usr/local/go/src/runtime/chan.go:388 +0x2b fp=0xc420053f50 sp=0xc420053f20
 pc=0x40439b
main.main()
  foo.go:9 +0x6f fp=0xc420053f80 sp=0xc420053f50 pc=0x4512ef
runtime.main()
  /usr/local/go/src/runtime/proc.go:185 +0x20d fp=0xc420053fe0 sp=0xc420053f80
 pc=0x424bad
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc420053fe8
 sp=0xc420053fe0 pc=0x44b4d1
```

```
goroutine 2 [force gc (idle)]:
runtime.gopark(0x4739b8, 0x4ad720, 0x47001e, 0xf, 0x14, 0x1)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003e768 sp=0xc42003e738
 pc=0x42503c
runtime.goparkunlock(0x4ad720, 0x47001e, 0xf, 0xc420000114, 0x1)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003e7a8 sp=0xc42003e768
 pc=0x42512e
runtime.forcegchelper()
  /usr/local/go/src/runtime/proc.go:238 +0xcc fp=0xc42003e7e0 sp=0xc42003e7a8
 pc=0x424e5c
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003e7e8
 sp=0xc42003e7e0 pc=0x44b4d1
created by runtime.init.4
  /usr/local/go/src/runtime/proc.go:227 +0x35

goroutine 3 [GC sweep wait]:
runtime.gopark(0x4739b8, 0x4ad7e0, 0x46fdd2, 0xd, 0x419914, 0x1)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003ef60 sp=0xc42003ef30
 pc=0x42503c
runtime.goparkunlock(0x4ad7e0, 0x46fdd2, 0xd, 0x14, 0x1)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003efa0 sp=0xc42003ef60
 pc=0x42512e
runtime.bgsweep(0xc42001e150)
  /usr/local/go/src/runtime/mgcsweep.go:52 +0xa3 fp=0xc42003efd8
 sp=0xc42003efa0 pc=0x419973
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003efe0
 sp=0xc42003efd8 pc=0x44b4d1
created by runtime.gcenable
  /usr/local/go/src/runtime/mgc.go:216 +0x58
one more line, no multiline
```

c. Enregistrez le fichier `test.log`.

4. Dans le dossier `multiline-app`, créez le Dockerfile.

   a. Collez le contenu suivant dans le fichier :

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
ADD test.log /test.log

RUN yum upgrade -y && yum install -y python3
```

```
WORKDIR /usr/local/bin

COPY main.py .

CMD ["python3", "main.py"]
```

b. Enregistrez le fichier `Dockerfile`.

5. À l'aide du Dockerfile, créez une image.

   a. Créez l'image : `docker build -t multiline-app-image`

      Où : `multiline-app-image` est le nom de l'image dans cet exemple.

   b. Vérifiez que l'image a été créée correctement : `docker images –filter reference=multiline-app-image`

      En cas de succès, la sortie montre l'image et l'identification `latest`.

6. Chargez l'image dans Amazon Elastic Container Registry.

   a. Créez un référentiel Amazon ECR pour stocker l'image : `aws ecr create-repository --repository-name multiline-app-repo --region us-east-1`

      Où : `multiline-app-repo` est le nom du référentiel et `us-east-1` est la région dans cet exemple.

      La sortie vous donne les détails du nouveau référentiel. Notez la valeur de `repositoryUri`, car vous en aurez besoin dans les étapes suivantes.

   b. Étiquetez votre image avec la valeur `repositoryUri` de la sortie précédente : `docker tag` *`multiline-app-image repositoryUri`*

      Exemple : `docker tag multiline-app-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

   c. Exécutez l'image docker pour vérifier qu'elle s'est exécutée correctement :`docker images –filter reference=`*`repositoryUri`*

      Dans la sortie, le nom du référentiel passe de `multiline-app-repo` à la valeur de `repositoryUri`.

   d. Envoyez (push) l'image vers Amazon ECR : `docker push` *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

Exemple : `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

Création de la définition de tâche et exécution de la tâche

1. Créez un fichier de définition de tâche avec le nom de fichier `multiline-task-definition.json`.
2. Collez le contenu suivant dans le fichier `multiline-task-definition.json` :

```
{
    "family": "firelens-example-multiline",
    "taskRoleArn": "task role ARN,
    "executionRoleArn": "execution role ARN",
    "containerDefinitions": [
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-
multiline-image:latest",
            "name": "log_router",
            "firelensConfiguration": {
                "type": "fluentbit",
                "options": {
                    "config-file-type": "file",
                    "config-file-value": "/extra.conf"
                }
            },
            "memoryReservation": 50
        },
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/multiline-app-
image:latest",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "cloudwatch_logs",
                    "region": "us-east-1",
                    "log_group_name": "multiline-test/application",
                    "auto_create_group": "true",
                    "log_stream_prefix": "multiline-"
```

```
                }
            },
            "memoryReservation": 100
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512"
}
```

Remplacez les éléments suivants dans la définition de tâche `multiline-task-definition.json` :

a. *`task role ARN`*

   Pour trouver l'ARN du rôle de la tâche, allez dans la console IAM. Choisissez Roles (Rôles) et trouvez le rôle de tâche `ecs-task-role-for-firelens` que vous avez créé. Choisissez le rôle et copiez l'ARN qui apparaît dans la section Summary (Résumé).

b. *`execution role ARN`*

   Pour trouver l'ARN du rôle d'exécution, allez dans la console IAM. Choisissez Roles (Rôles) et trouvez le rôle `ecsTaskExecutionRole`. Choisissez le rôle et copiez l'ARN qui apparaît dans la section Summary (Résumé).

c. *`aws_account_id`*

   Pour trouver votre `aws_account_id`, connectez-vous à la AWS Management Console. Choisissez votre nom d'utilisateur en haut à droite et copiez votre ID de compte.

d. *`us-east-1`*

   Remplacez la région si nécessaire.

3. Enregistrez le fichier de définition de tâche : `aws ecs register-task-definition --cli-input-json file://multiline-task-definition.json --region us-east-1`
4. Ouvrez la console à la https://console.aws.amazon.com/ecs/version 2.
5. Dans le panneau de navigation, choisissez Task Definitions (Définitions de tâches), puis la famille `firelens-example-multiline`, car nous avons enregistré la définition de tâche à cette famille dans la première ligne de la définition de tâche ci-dessus.
6. Choisissez la dernière version.

7. Choisissez Déployer, Exécuter la tâche.
8. Sur la page Exécuter la tâche, pour Cluster, choisissez le cluster, puis sous Mise en réseau, pour Sous-réseaux, choisissez les sous-réseaux disponibles pour votre tâche.
9. Choisissez Créer.

Vérifiez que les messages de journal multilignes dans Amazon CloudWatch apparaissent concaténés

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, développez Logs (Journaux) et choisissez Log groups (Groupes de journaux).
3. Choisissez le groupe de journaux `multiline-test/applicatio`.
4. Choisissez le journal et affichez les messages. Les lignes qui correspondent aux règles du fichier analyseur sont concaténées et apparaissent comme un seul message.

   L'extrait de journal suivant montre un suivi de pile Go qui est concaténé en un seul événement :

```
{
    "log": "panic: my panic\n\ngoroutine 4 [running]:\npanic(0x45cb40,
 0x47ad70)\n  /usr/local/go/src/runtime/panic.go:542 +0x46c fp=0xc42003f7b8
 sp=0xc42003f710 pc=0x422f7c\nmain.main.func1(0xc420024120)\n  foo.go:6
 +0x39 fp=0xc42003f7d8 sp=0xc42003f7b8 pc=0x451339\nruntime.goexit()\n  /usr/
local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003f7e0 sp=0xc42003f7d8
 pc=0x44b4d1\ncreated by main.main\n  foo.go:5 +0x58\n\ngoroutine 1 [chan receive]:
\nruntime.gopark(0x4739b8, 0xc420024178, 0x46fcd7, 0xc, 0xc420028e17, 0x3)\n  /usr/
local/go/src/runtime/proc.go:280 +0x12c fp=0xc420053e30 sp=0xc420053e00 pc=0x42503c
\nruntime.goparkunlock(0xc420024178, 0x46fcd7, 0xc, 0x1000f010040c217, 0x3)\n
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc420053e70 sp=0xc420053e30
 pc=0x42512e\nruntime.chanrecv(0xc420024120, 0x0, 0xc420053f01, 0x4512d8)\n
  /usr/local/go/src/runtime/chan.go:506 +0x304 fp=0xc420053f20 sp=0xc420053e70
 pc=0x4046b4\nruntime.chanrecv1(0xc420024120, 0x0)\n  /usr/local/go/src/runtime/
chan.go:388 +0x2b fp=0xc420053f50 sp=0xc420053f20 pc=0x40439b\nmain.main()\n
 foo.go:9 +0x6f fp=0xc420053f80 sp=0xc420053f50 pc=0x4512ef\nruntime.main()\n
  /usr/local/go/src/runtime/proc.go:185 +0x20d fp=0xc420053fe0 sp=0xc420053f80
 pc=0x424bad\nruntime.goexit()\n  /usr/local/go/src/runtime/asm_amd64.s:2337
 +0x1 fp=0xc420053fe8 sp=0xc420053fe0 pc=0x44b4d1\n\ngoroutine 2 [force gc
 (idle)]:\nruntime.gopark(0x4739b8, 0x4ad720, 0x47001e, 0xf, 0x14, 0x1)\n  /
usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003e768 sp=0xc42003e738
 pc=0x42503c\nruntime.goparkunlock(0x4ad720, 0x47001e, 0xf, 0xc420000114, 0x1)\n
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003e7a8 sp=0xc42003e768
 pc=0x42512e\nruntime.forcegchelper()\n  /usr/local/go/src/runtime/proc.go:238
```

```
 +0xcc fp=0xc42003e7e0 sp=0xc42003e7a8 pc=0x424e5c\nruntime.goexit()\n  /usr/
local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003e7e8 sp=0xc42003e7e0
 pc=0x44b4d1\ncreated by runtime.init.4\n  /usr/local/go/src/runtime/proc.go:227
 +0x35\n\ngoroutine 3 [GC sweep wait]:\nruntime.gopark(0x4739b8, 0x4ad7e0,
 0x46fdd2, 0xd, 0x419914, 0x1)\n  /usr/local/go/src/runtime/proc.go:280 +0x12c
 fp=0xc42003ef60 sp=0xc42003ef30 pc=0x42503c\nruntime.goparkunlock(0x4ad7e0,
 0x46fdd2, 0xd, 0x14, 0x1)\n  /usr/local/go/src/runtime/proc.go:286 +0x5e
 fp=0xc42003efa0 sp=0xc42003ef60 pc=0x42512e\nruntime.bgsweep(0xc42001e150)\n
  /usr/local/go/src/runtime/mgcsweep.go:52 +0xa3 fp=0xc42003efd8 sp=0xc42003efa0
 pc=0x419973\nruntime.goexit()\n  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1
 fp=0xc42003efe0 sp=0xc42003efd8 pc=0x44b4d1\ncreated by runtime.gcenable\n  /usr/
local/go/src/runtime/mgc.go:216 +0x58",
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:2"
}
```

L'extrait de journal suivant montre comment le même événement apparaît si vous exécutez un conteneur ECS qui n'est pas configuré pour concaténer les messages de journaux multilignes. Le champ du journal contient une seule ligne.

```
{
    "log": "panic: my panic",
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:3"
```

> **Note**
>
> Si vos journaux sont envoyés dans des fichiers journaux au lieu de la sortie standard, nous vous recommandons de spécifier les paramètres de configuration `multiline.parser` et `multiline.key_content` dans le plugin d'entrée Tail au lieu du filtre.

Déploiement de Fluent Bit sur des conteneurs Windows Amazon ECS

Fluent Bit est un processeur et un routeur de journaux rapides et flexibles pris en charge par différents systèmes d'exploitation. Il peut être utilisé pour acheminer les journaux vers diverses AWS destinations telles qu'Amazon CloudWatch Logs, Firehose, Amazon S3 et Amazon OpenSearch Service. Fluent Bit prend en charge des solutions partenaires courantes, comme Datadog, Splunk et les serveurs HTTP personnalisés. Pour plus d'informations sur Fluent Bit, consultez le site web Fluent Bit .

L'image AWS pour Fluent Bit est disponible sur Amazon ECR à la fois sur la galerie publique Amazon ECR et dans un référentiel Amazon ECR dans la plupart des Régions pour une haute disponibilité. Pour plus d'informations, consultez aws-for-fluent-bit le GitHub site Web.

Ce didacticiel explique comment déployer des conteneurs Fluent Bit sur leurs instances Windows exécutées dans Amazon ECS afin de diffuser les journaux générés par les tâches Windows vers Amazon CloudWatch pour une journalisation centralisée.

Ce didacticiel utilise l'approche suivante :

- Fluent Bit fonctionne comme un service avec la stratégie de planification du démon. Cette stratégie garantit qu'une seule instance de Fluent Bit s'exécute toujours sur les instances de conteneur du cluster.
  - Écoute sur le port 24224 à l'aide du plug-in d'entrée directe.
  - Expose le port 24224 à l'hôte afin que le moteur d'exécution du docker puisse envoyer des journaux à Fluent Bit en utilisant ce port exposé.
  - Dispose d'une configuration qui permet à Fluent Bit d'envoyer les enregistrements des journaux vers des destinations spécifiées.
- Lancement de tous les autres conteneurs de tâches Amazon ECS à l'aide du pilote de journalisation Fluentd. Pour plus d'informations, consultez Fluentd logging drive (Pilote de journalisation Fluentd) sur le site Web de la documentation de Docker.
  - Docker se connecte au socket TCP 24224 sur localhost dans l'espace de noms de l'hôte.
  - L'agent Amazon ECS ajoute des étiquettes aux conteneurs, notamment le nom du cluster, le nom de famille de la définition de la tâche, le numéro de révision de la définition de la tâche, l'ARN de la tâche et le nom du conteneur. Les mêmes informations sont ajoutées à l'enregistrement du journal à l'aide de l'option labels du pilote de journalisation fluentd de Docker.

Pour de plus amples informations, veuillez consulter labels, labels-regex, env, and env-regex sur le site web de la documentation de Docker.

- Comme l'option `async` du pilote de journalisation Fluentd est définie sur `true`, lorsque le conteneur Fluent Bit est redémarré, Docker met les journaux en mémoire tampon jusqu'à ce que le conteneur Fluent Bit soit redémarré. Vous pouvez augmenter la limite de mémoire tampon en définissant fluentd-buffer-limit cette option. Pour plus d'informations, consultez fluentd-buffer-limitle site Web de documentation de Docker.

Le flux de travail est le suivant :

- Le conteneur Fluent Bit démarre et écoute sur le port 24224 qui est exposé à l'hôte.
- Fluent Bit utilise les informations d'identification du rôle IAM de la tâche spécifiées dans sa définition de tâche.
- Les autres tâches lancées sur la même instance utilisent le pilote de journalisation fluentd de Docker pour se connecter au conteneur Fluent Bit sur le port 24224.
- Lorsque les conteneurs d'applications génèrent des journaux, le moteur d'exécution de Docker étiquette ces enregistrements, ajoute des métadonnées supplémentaires spécifiées dans les étiquettes, puis les transmet sur le port 24224 de l'espace de noms de l'hôte.
- Fluent Bit reçoit l'enregistrement du journal sur le port 24224 car il est exposé à l'espace de noms de l'hôte.
- Fluent Bit effectue son traitement interne et achemine les journaux comme spécifié.

Ce didacticiel utilise la configuration CloudWatch Fluent Bit par défaut qui effectue les opérations suivantes :

- Crée un nouveau groupe de journaux pour chaque cluster et chaque famille de définitions de tâches.
- Crée un nouveau flux de journaux pour chaque conteneur de tâches du groupe de journaux généré ci-dessus chaque fois qu'une nouvelle tâche est lancée. Chaque flux sera marqué avec l'identifiant de tâche à laquelle le conteneur appartient.
- Ajoute des métadonnées supplémentaires, notamment le nom du cluster, l'ARN de la tâche, le nom du conteneur de la tâche, la famille de définitions de la tâche et le numéro de révision de la définition de la tâche dans chaque entrée du journal.

Par exemple, si vous avez `task_1` with `container_1` `container_2` et t `ask_2` with`container_3`, les flux de CloudWatch log sont les suivants :

- `/aws/ecs/windows.ecs_task_1`

  `task-out.`*`TASK_ID`*`.container_1`

  `task-out.`*`TASK_ID`*`.container_2`
- `/aws/ecs/windows.ecs_task_2`

  `task-out.`*`TASK_ID`*`.container_3`

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir de AWS CLI SDKs, et de l'API Amazon ECS via les deux IPv4 et. IPv6 Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

Étapes

- Prérequis
- Étape 1 : Créer les rôles IAM d'accès
- Étape 2 : Créer une instance de conteneur Windows Amazon ECS
- Étape 3 : Configurer Fluent Bit
- Étape 4 : enregistrer une définition de tâche Windows Fluent Bit qui achemine les journaux vers CloudWatch
- Étape 5 : Exécuter la définition de tâche ecs-windows-fluent-bit en tant que service Amazon ECS en utilisant la stratégie de planification du démon
- Étape 6 : Enregistrer une définition de tâche Windows qui génère les journaux
- Étape 7 : Créer la définition de tâche windows-app-task
- Étape 8 : Vérifiez les connexions CloudWatch
- Étape 9 : nettoyer

Prérequis

Le didacticiel suppose de remplir les prérequis suivants :

- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations, voir Installation ou mise à jour vers la dernière version du AWS CLI.
- L'image du conteneur `aws-for-fluent-bit` est disponible pour les systèmes d'exploitation Windows suivants :
  - Windows Server 2019 Core
  - Windows Server 2019 Full
  - Windows Server 2022 Core
  - Windows Server 2022 Full
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Vous avez un cluster. Dans ce didacticiel, le nom du cluster est FluentBit-cluster.
- Vous disposez d'un VPC doté d'un sous-réseau public dans lequel l' EC2 instance sera lancée. Vous pouvez utiliser votre VPC par défaut. Vous pouvez également utiliser un sous-réseau privé qui permet aux CloudWatch points de terminaison Amazon d'accéder au sous-réseau. Pour plus d'informations sur les CloudWatch points de terminaison Amazon, consultez la section CloudWatch Points de terminaison et quotas Amazon dans le. Références générales AWS Pour plus d'informations sur la manière d'utiliser l'assistant Amazon VPC pour créer un VPC, consultez the section called “Créer un Virtual Private Cloud”.

Étape 1 : Créer les rôles IAM d'accès

Créez les rôles IAM Amazon ECS.

1. Créez le rôle d'instance de conteneur Amazon ECS nommé « ecsInstanceRole ». Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.
2. Créez un rôle IAM pour la tâche Fluent Bit intitulée `fluentTaskRole`. Pour de plus amples informations, veuillez consulter the section called “Rôle IAM de tâche”.

   Les autorisations IAM accordées dans ce rôle IAM sont prises en charge par les conteneurs de tâches. Pour autoriser Fluent Bit à envoyer des journaux CloudWatch, vous devez associer les autorisations suivantes au rôle IAM de la tâche.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:CreateLogGroup",
            "logs:DescribeLogStreams",
            "logs:PutLogEvents"
        ],
        "Resource": "*"
    }
    ]
}
```

3. Attachez la stratégie au rôle.

   a. Enregistrez le contenu ci-dessus dans un fichier nommé `fluent-bit-policy.json`.

   b. Exécutez la commande suivante pour associer la politique en ligne au rôle IAM `fluentTaskRole`.

```
aws iam put-role-policy --role-name fluentTaskRole --policy-name
 fluentTaskPolicy --policy-document file://fluent-bit-policy.json
```

Étape 2 : Créer une instance de conteneur Windows Amazon ECS

Créez une instance de conteneur Windows Amazon ECS.

Pour créer une instance Amazon ECS

1. Utilisez la commande `aws ssm get-parameters` pour récupérer l'ID AMI de la région qui héberge votre VPC. Pour plus d'informations, consultez Extraction des métadonnées d'AMI optimisée pour Amazon ECS.
2. Utilisez la EC2 console Amazon pour lancer l'instance.

   a. Ouvrez la EC2 console Amazon à l'adresse https://console.aws.amazon.com/ec2/.

b. Dans la barre de navigation, sélectionnez la région à utiliser.

c. Dans le EC2 tableau de bord, choisissez Launch instance.

d. Pour Name (Nom), saisissez un nom unique.

e. Pour Application and OS Images (Amazon Machine Image) (Images de l'application et du SE (Amazon Machine Image)), choisissez l'AMI que vous avez récupérée lors de la première étape.

f. Concernant l'option Instance type (Type d'instance), choisissez `t3.xlarge`.

g. Pour Key pair (login) (Paire de clés (connexion)), choisissez une paire de clés.

h. Sous Network Settings (Paramètres réseau), pour Security group (Groupe de sécurité), choisissez un groupe de sécurité existant ou créez-en un nouveau.

i. Sous Network settings (Paramètres réseau), pour Auto-assign Public IP (Attribuer automatiquement l'adresse IP publique), sélectionnez Enable (Activer).

j. Sous Détails avancés, pour le profil d'instance IAM, sélectionnez ecsInstanceRole.

k. Configurez votre instance de conteneur Amazon ECS avec les données utilisateur suivantes. Sous Détails avancés, collez le script suivant dans le champ Données utilisateur, en le *`cluster_name`* remplaçant par le nom de votre cluster.

```
<powershell>
Import-Module ECSTools
Initialize-ECSAgent -Cluster cluster-name -EnableTaskENI -EnableTaskIAMRole -
LoggingDrivers '["awslogs","fluentd"]'
</powershell>
```

l. Lorsque vous êtes prêt, cochez la case de confirmation, puis sélectionnez Launch Instances (Lancer des instances).

m. Une page de confirmation indique que l'instance est en cours de lancement. Sélectionnez View Instances (Afficher les instances) pour fermer la page de confirmation et revenir à la console.

Étape 3 : Configurer Fluent Bit

Vous pouvez utiliser la configuration par défaut suivante fournie par AWS pour démarrer rapidement :

- Amazon CloudWatch, qui est basé sur le plug-in Fluent Bit pour Amazon figurant dans CloudWatch le manuel officiel de Fluent Bit.

Vous pouvez également utiliser d'autres configurations par défaut fournies par AWS. Pour plus d'informations, consultez la section Overriding the entrypoint for the Windows image (Ignorer le point d'entrée de l'image Windows) sur le site Web `aws-for-fluent-bit` de Github.

La configuration par défaut d'Amazon CloudWatch Fluent Bit est illustrée ci-dessous.

Remplacez les variables suivantes :

- *`region`*avec la région dans laquelle vous souhaitez envoyer les CloudWatch journaux Amazon.

```
[SERVICE]
    Flush               5
    Log_Level           info
    Daemon              off

[INPUT]
    Name                forward
    Listen              0.0.0.0
    Port                24224
    Buffer_Chunk_Size   1M
    Buffer_Max_Size     6M
    Tag_Prefix          ecs.

# Amazon ECS agent adds the following log keys as labels to the docker container.
# We would use fluentd logging driver to add these to log record while sending it to
 Fluent Bit.
[FILTER]
    Name                modify
    Match               ecs.*
    Rename              com.amazonaws.ecs.cluster ecs_cluster
    Rename              com.amazonaws.ecs.container-name ecs_container_name
    Rename              com.amazonaws.ecs.task-arn ecs_task_arn
    Rename              com.amazonaws.ecs.task-definition-family
 ecs_task_definition_family
    Rename              com.amazonaws.ecs.task-definition-version
 ecs_task_definition_version

[FILTER]
    Name                rewrite_tag
    Match               ecs.*
    Rule                $ecs_task_arn ^([a-z-:0-9]+)/([a-zA-Z0-9-_]+)/([a-z0-9]+)$
 out.$3.$ecs_container_name false
```

```
    Emitter_Name        re_emitted

[OUTPUT]
    Name                cloudwatch_logs
    Match               out.*
    region              region
    log_group_name      fallback-group
    log_group_template  /aws/ecs/$ecs_cluster.$ecs_task_definition_family
    log_stream_prefix   task-
    auto_create_group   On
```

Chaque journal qui entre dans Fluent Bit possède une étiquette que vous spécifiez. Si vous n'en fournissez pas, elle est générée automatiquement. Les étiquettes peuvent être utilisées pour acheminer différents journaux vers différentes destinations. Pour plus d'informations, voir Tag (Étiquette) dans le manuel officiel de Fluent Bit.

La configuration Fluent Bit décrite ci-dessus possède les propriétés suivantes :

- Le plug-in d'entrée directe écoute le trafic entrant sur le port TCP 24224.
- Chaque entrée de journal reçue sur ce port possède une étiquette que le plug-in d'entrée directe modifie pour préfixer l'enregistrement avec une chaîne `ecs.`.
- Le pipeline interne de Fluent Bit achemine l'entrée du journal pour modifier le filtre à l'aide d'une correspondance d'expression régulière (regex). Ce filtre remplace les clés dans l'enregistrement de journal au format JSON par le format que Fluent Bit peut utiliser.
- L'entrée de journal modifiée est ensuite utilisée par le filtre rewrite_tag. Ce filtre remplace la balise de l'enregistrement du journal par le format out. *TASK_ID*. *CONTAINER_NAME*.
- La nouvelle balise sera acheminée vers le plug-in de sortie cloudwatch_logs qui crée les groupes de journaux et les flux comme décrit précédemment en utilisant les `log_stream_prefix` options `log_group_template` et du plug-in de sortie. CloudWatch Pour plus d'informations, voir Configuration parameters (Paramètres de configuration) dans le manuel officiel de Fluent Bit.

Étape 4 : enregistrer une définition de tâche Windows Fluent Bit qui achemine les journaux vers CloudWatch

Enregistrez une définition de tâche Windows Fluent Bit vers laquelle les journaux sont acheminés CloudWatch.

> **Note**
>
> Cette définition de tâche expose le port 24224 du conteneur Fluent Bit au port hôte 24224. Vérifiez que ce port n'est pas ouvert dans le groupe de sécurité de votre EC2 instance pour empêcher l'accès depuis l'extérieur.

Pour enregistrer une définition de tâche

1. Créez un fichier nommé `fluent-bit.json` avec les contenus suivants.

   Remplacez les variables suivantes :

   - *`task-iam-role`*avec le nom de ressource Amazon (ARN) de votre tâche (rôle IAM)
   - *`region`*avec la région dans laquelle s'exécute votre tâche

```
{
  "family": "ecs-windows-fluent-bit",
  "taskRoleArn": "task-iam-role",
  "containerDefinitions": [
    {
      "name": "fluent-bit",
      "image": "public.ecr.aws/aws-observability/aws-for-fluent-
bit:windowsservercore-latest",
      "cpu": 512,
      "portMappings": [
        {
          "hostPort": 24224,
          "containerPort": 24224,
          "protocol": "tcp"
        }
      ],
      "entryPoint": [
        "Powershell",
        "-Command"
      ],
      "command": [
        "C:\\entrypoint.ps1 -ConfigFile C:\\ecs_windows_forward_daemon\
\cloudwatch.conf"
      ],
      "environment": [
```

```
        {
          "name": "AWS_REGION",
          "value": "region"
        }
      ],
      "memory": 512,
      "essential": true,
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/fluent-bit-logs",
          "awslogs-region": "region",
          "awslogs-stream-prefix": "flb",
          "awslogs-create-group": "true"
        }
      }
    }
  ],
  "memory": "512",
  "cpu": "512"
}
```

2. Utilisez la commande suivante pour enregistrer la définition de tâche.

```
aws ecs register-task-definition --cli-input-json file://fluent-bit.json --
region region
```

Vous pouvez répertorier les définitions de tâche de votre compte en exécutant la commande `list-task-definitions`. La sortie affiche les valeurs de famille et de révision que vous pouvez utiliser conjointement avec `run-task` ou `start-task`.

Étape 5 : Exécuter la définition de tâche **`ecs-windows-fluent-bit`** en tant que service Amazon ECS en utilisant la stratégie de planification du démon

Après avoir enregistré une définition de tâche pour votre compte, vous pouvez exécuter une tâche dans le cluster. Pour ce didacticiel, vous exécutez une instance de la définition de tâche `ecs-windows-fluent-bit:1` de votre cluster `FluentBit-cluster`. Exécutez la tâche dans un service qui utilise la stratégie de planification du démon, qui garantit qu'une seule instance de Fluent Bit s'exécute toujours sur chacune de vos instances de conteneur.

Pour exécuter une tâche

1. Exécutez la commande suivante pour démarrer la définition de la tâche `ecs-windows-fluent-bit:1` (enregistrée à l'étape précédente) en tant que service.

   > **Note**
   >
   > Cette définition de tâche utilise le pilote de journalisation `awslogs`. Votre instance de conteneur doit disposer des autorisations nécessaires.

   Remplacez les variables suivantes :

   - *`region`*avec la région où fonctionne votre service

   ```
   aws ecs create-service \
       --cluster FluentBit-cluster \
       --service-name FluentBitForwardDaemonService \
       --task-definition ecs-windows-fluent-bit:1 \
       --launch-type EC2 \
       --scheduling-strategy DAEMON \
       --region region
   ```

2. Exécutez la commande suivante pour répertorier vos tâches.

   Remplacez les variables suivantes :

   - *`region`*avec la région dans laquelle s'exécutent vos tâches de service

   ```
   aws ecs list-tasks --cluster FluentBit-cluster --region region
   ```

Étape 6 : Enregistrer une définition de tâche Windows qui génère les journaux

Enregistrez une définition de tâche qui génère les journaux. Cette définition de tâche déploie une image de conteneur Windows qui écrira un nombre incrémentiel à `stdout` toutes les secondes.

La définition de la tâche utilise le pilote de journalisation Fluentd qui se connecte au port 24224 que le plug-in Fluent Bit écoute. L'agent Amazon ECS met des étiquettes à chaque conteneur ECS qui incluent le nom du cluster, l'ARN de la tâche, le nom de famille de la définition de la tâche, le numéro de révision de la définition de la tâche et le nom du conteneur de la tâche. Ces étiquettes clé-valeur sont transmises à Fluent Bit.

> **Note**
>
> Cette tâche utilise également le mode réseau `default`. Toutefois, vous pouvez aussi utiliser le mode réseau `awsvpc` avec la tâche.

Pour enregistrer une définition de tâche

1. Créez un fichier nommé `windows-app-task.json` avec les contenus suivants.

```
{
  "family": "windows-app-task",
  "containerDefinitions": [
    {
      "name": "sample-container",
      "image": "mcr.microsoft.com/windows/servercore:ltsc2019",
      "cpu": 512,
      "memory": 512,
      "essential": true,
      "entryPoint": [
        "Powershell",
        "-Command"
      ],
      "command": [
        "$count=1;while(1) { Write-Host $count; sleep 1; $count=$count+1;}"
      ],
      "logConfiguration": {
        "logDriver": "fluentd",
        "options": {
          "fluentd-address": "localhost:24224",
          "tag": "{{ index .ContainerLabels \"com.amazonaws.ecs.task-definition-
family\" }}",
          "fluentd-async": "true",
          "labels": "com.amazonaws.ecs.cluster,com.amazonaws.ecs.container-
name,com.amazonaws.ecs.task-arn,com.amazonaws.ecs.task-definition-
family,com.amazonaws.ecs.task-definition-version"
```

```
        }
      }
    }
  ],
  "memory": "512",
  "cpu": "512"
}
```

2. Utilisez la commande suivante pour enregistrer la définition de tâche.

   Remplacez les variables suivantes :

   - *region*avec la région dans laquelle s'exécute votre tâche

```
aws ecs register-task-definition --cli-input-json file://windows-app-task.json --
region region
```

   Vous pouvez répertorier les définitions de tâche de votre compte en exécutant la commande `list-task-definitions`. La sortie affiche les valeurs de famille et de révision que vous pouvez utiliser conjointement avec `run-task` ou `start-task`.

Étape 7 : Créer la définition de tâche **`windows-app-task`**

Après avoir enregistré la définition de tâche `windows-app-task`, exécutez-la dans votre cluster `FluentBit-cluster`.

Pour exécuter une tâche

1. Exécutez la définition de tâche `windows-app-task:1` que vous avez enregistrée à l'étape précédente.

   Remplacez les variables suivantes :

   - *region*avec la région dans laquelle s'exécute votre tâche

```
aws ecs run-task --cluster FluentBit-cluster --task-definition windows-app-task:1
 --count 2 --region region
```

2. Exécutez la commande suivante pour répertorier vos tâches.

```
aws ecs list-tasks --cluster FluentBit-cluster
```

Étape 8 : Vérifiez les connexions CloudWatch

Afin de vérifier votre configuration Fluent Bit, vérifiez les groupes de journaux suivants dans la CloudWatch console :

- `/ecs/fluent-bit-logs` : il s'agit du groupe de journaux qui correspond au conteneur de démon Fluent Bit qui s'exécute sur l'instance de conteneur.
- `/aws/ecs/FluentBit-cluster.windows-app-task` : il s'agit du groupe de journaux qui correspond à toutes les tâches lancées pour la famille de définition de tâche `windows-app-task` au sein du cluster `FluentBit-cluster`.

  `task-out.FIRST_TASK_ID.sample-container` : ce flux de journaux contient tous les journaux générés par la première instance de la tâche dans le conteneur de tâches sample-container.

  `task-out.SECOND_TASK_ID.sample-container` : ce flux de journaux contient tous les journaux générés par la seconde instance de la tâche dans le conteneur de tâches sample-container.

Le flux de journaux `task-out.TASK_ID.sample-container` contient des champs similaires aux suivants :

```
{
    "source": "stdout",
    "ecs_task_arn": "arn:aws:ecs:region:0123456789012:task/FluentBit-
cluster/13EXAMPLE",
    "container_name": "/ecs-windows-app-task-1-sample-container-cEXAMPLE",
    "ecs_cluster": "FluentBit-cluster",
    "ecs_container_name": "sample-container",
    "ecs_task_definition_version": "1",
    "container_id": "61f5e6EXAMPLE",
    "log": "10",
    "ecs_task_definition_family": "windows-app-task"
}
```

Pour vérifier la configuration Fluent Bit

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, choisissez Groupes de journaux. Assurez-vous que vous êtes dans la région où vous avez déployé Fluent Bit sur vos conteneurs.

   Dans la liste des groupes de journaux du Région AWS, vous devriez voir ce qui suit :

   - `/ecs/fluent-bit-logs`
   - `/aws/ecs/FluentBit-cluster.windows-app-task`

   Si vous voyez ces groupes de journaux, la vérification de la configuration Fluent Bit est terminée.

Étape 9 : nettoyer

Lorsque vous avez terminé ce didacticiel, nettoyez les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources que vous n'utilisez pas.

Pour nettoyer les ressources du didacticiel

1. Arrêtez la tâche `windows-simple-task` et la tâche `ecs-fluent-bit`. Pour de plus amples informations, veuillez consulter the section called “Arrêt d'une tâche”.
2. Utilisez la commande suivante pour supprimer le groupe de journaux `/ecs/fluent-bit-logs`. Pour plus d'informations, sur la suppression de groupes de journaux delete-log-group, voir la AWS Command Line Interface référence.

   ```
   aws logs delete-log-group --log-group-name /ecs/fluent-bit-logs
   aws logs delete-log-group --log-group-name /aws/ecs/FluentBit-cluster.windows-app-
   task
   ```

3. Exécutez la commande suivante pour mettre fin à l'instance.

   ```
   aws ec2 terminate-instances --instance-ids instance-id
   ```

4. Exécutez les commandes suivantes pour supprimer les rôles IAM.

   ```
   aws iam delete-role --role-name ecsInstanceRole
   aws iam delete-role --role-name fluentTaskRole
   ```

5. Exécutez la commande suivante pour supprimer le cluster Amazon ECS.

```
aws ecs delete-cluster --cluster FluentBit-cluster
```

Utilisation gMSA pour les EC2 Linux conteneurs sur Amazon ECS

Amazon ECS prend en charge l'authentification Active Directory pour les conteneurs Linux EC2 via un type spécial de compte de service appelé compte de service géré de groupe (gMSA).

Les applications réseau Linux, comme les applications .NET Core, peuvent faire appel à Active Directory pour faciliter la gestion de l'authentification et des autorisations entre les utilisateurs et les services. Vous pouvez utiliser cette fonctionnalité en concevant des applications qui s'intègrent à Active Directory et s'exécutent sur des serveurs joints à un domaine. Toutefois, comme les conteneurs Linux ne peuvent pas être joints à un domaine, vous devez configurer un conteneur Linux pour qu'il soit exécuté avec gMSA.

Un Linux conteneur qui s'exécute avec gMSA repose sur le `credentials-fetcher` daemon qui s'exécute sur l' EC2 instance Amazon hôte du conteneur. En d'autres termes, le démon récupère les informations d'identification gMSA auprès du contrôleur de domaine Active Directory, puis les transfère à l'instance de conteneur. Pour plus d'informations sur les comptes de service, veuillez consulter Créer des gMSAs pour des conteneurs Windows sur le site Web de Microsoft Learn.

Considérations

Tenez compte des points suivants avant d'utiliser des gMSA pour les conteneurs Linux :

- Si vos conteneurs fonctionnent encore EC2, vous pouvez les utiliser gMSA pour les Windows conteneurs et les Linux conteneurs. Pour plus d'informations sur l'utilisation gMSA du conteneur Linux sur Fargate, consultez. Utilisation gMSA pour les Linux conteneurs sur Fargate
- Vous aurez peut-être besoin d'un ordinateur Windows joint au domaine pour remplir les conditions requises. Par exemple, vous pourriez avoir besoin d'un ordinateur Windows joint au domaine pour créer le gMSA dans Active Directory avec PowerShell. Les PowerShell outils RSAT Active Director ne sont disponibles que pourWindows. Pour plus d'informations, veuillez consulter Installation des outils d'administration Active Directory (langue française non garantie).
- Vous avez choisi entre les gMSA sans domaine et joindre chaque instance à un seul domaine. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres

applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.

Choisissez ensuite le stockage de données pour CredSpec et, éventuellement, pour les informations d'identification utilisateur Active Directory pour les gMSA sans domaine.

Amazon ECS utilise un fichier de spécification des informations d'identification Active Directory (CredSpec). Ce fichier contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur. Vous générez le fichier CredSpec, puis vous le stockez dans l'une des options de stockage CredSpec du tableau suivant, spécifique au système d'exploitation des instances de conteneur. Pour utiliser la méthode sans domaine, une section facultative du fichier CredSpec peut spécifier les informations d'identification dans l'une des options de stockage domainless user credentials du tableau suivant, spécifiques au système d'exploitation des instances de conteneur.

Emplacement de stockage	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	informations d'identification utilisateur sans domaine	informations d'identification utilisateur sans domaine
Boutique de paramètre s Amazon EC2 Systems Manager	CredSpec	CredSpec, informations d'identification utilisateur sans domaine
Fichier local	N/A	CredSpec

Prérequis

Avant d'utiliser la fonctionnalité gMSA pour les conteneurs Linux avec Amazon ECS, assurez-vous de remplir les conditions suivantes :

- Vous configurez un domaine Active Directory avec les ressources auxquelles vous souhaitez que vos conteneurs accèdent. Amazon ECS prend en charge les configurations suivantes :
  - Un AWS Directory Service Active Directory. AWS Directory Service est un Active Directory AWS géré hébergé sur Amazon EC2. Pour plus d'informations, consultez Getting Started with AWS Managed Microsoft AD dans le Guide d'AWS Directory Service administration.

  - Un répertoire Active Directory sur site. Vous devez vous assurer que l'instance de conteneur Linux Amazon ECS peut se joindre au domaine. Pour de plus amples informations, veuillez consulter AWS Direct Connect.
- Vous disposez d'un compte gMSA existant dans l'Active Directory. Pour de plus amples informations, veuillez consulter Utilisation gMSA pour les EC2 Linux conteneurs sur Amazon ECS.
- Vous avez installé et vous exécutez le démon `credentials-fetcher` sur une instance de conteneur Linux Amazon ECS. Vous avez également ajouté un jeu initial d'informations d'identification au démon `credentials-fetcher` pour vous authentifier auprès d'Active Directory.

> **Note**
>
> Le démon `credentials-fetcher` est uniquement disponible pour Amazon Linux 2023 et Fedora 37 et versions ultérieures. Le démon n'est pas disponible pour Amazon Linux 2. Pour plus d'informations, consultez aws/credentials-fetcher on. GitHub

- Vous configurez les informations d'identification permettant au démon `credentials-fetcher` de s'authentifier auprès d'Active Directory. Les informations d'identification doivent être membres du groupe de sécurité Active Directory qui a accès au compte gMSA. Il existe plusieurs options dans Décidez si vous souhaitez joindre les instances au domaine ou utiliser le gMSA sans domaine..
- Vous avez ajouté les autorisations IAM requises. Les autorisations requises dépendent des méthodes que vous choisissez pour les informations d'identification initiales et pour le stockage de la spécification des informations d'identification :
  - Si vous utilisez domainless gMSA pour les informations d'identification initiales, les autorisations IAM pour AWS Secrets Manager sont requises pour le rôle d'exécution de la tâche.
  - Si vous stockez la spécification des informations d'identification dans SSM Parameter Store, les autorisations IAM pour Amazon EC2 Systems Manager Parameter Store sont requises pour le rôle d'exécution de la tâche.
  - Si vous stockez la spécification des informations d'identification dans Amazon S3, les autorisations IAM d'Amazon Simple Storage Service sont requises sur le rôle d'exécution de la tâche.

Configuration de conteneurs Linux compatibles avec gMSA sur Amazon ECS

Préparation de l'infrastructure

Les étapes suivantes sont des considérations et une configuration qui ne sont effectuées qu'une seule fois. Après avoir terminé ces étapes, vous pouvez automatiser la création d'instances de conteneur afin de réutiliser cette configuration.

Décidez de la manière dont les informations d'identification initiales sont fournies et configurez les données EC2 utilisateur dans un modèle de EC2 lancement réutilisable pour installer le `credentials-fetcher` daemon.

1. Décidez si vous souhaitez joindre les instances au domaine ou utiliser le gMSA sans domaine.
   - Joindre EC2 des instances au domaine Active Directory
     - Jointure des instances par les données utilisateur

       Ajoutez les étapes permettant de joindre le domaine Active Directory à vos données EC2 utilisateur dans un modèle de EC2 lancement. Plusieurs groupes Amazon EC2 Auto Scaling peuvent utiliser le même modèle de lancement.

       Vous pouvez suivre ces étapes pour Rejoindre un Active Directory ou un domaine FreeIPA dans les Fedora Docs (langue française non garantie).
   - Création d'un utilisateur Active Directory pour gMSA sans domaine

     Le démon `credentials-fetcher` possède une fonctionnalité appelée gMSA sans domaine. Cette fonctionnalité nécessite un domaine, mais il n'est pas nécessaire que l' EC2instance soit jointe au domaine. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance. Vous devez plutôt fournir le nom d'un secret d' AWS Secrets Manager dans le fichier CredSpec. Le secret doit contenir un nom d'utilisateur, un mot de passe et le domaine auquel se connecter.

     Cette fonctionnalité est prise en charge et peut être utilisée avec les conteneurs Linux et Windows.

Cette fonctionnalité est similaire à la fonctionnalité gMSA support for non-domain-joined container hosts. Pour plus d'informations sur la fonctionnalité Windows, veuillez consulter Architecture et améliorations des gMSA sur le site Web de Microsoft Learn.

a. Créez un utilisateur dans votre domaine Active Directory. L'utilisateur d'Active Directory doit être autorisé à accéder aux comptes de service gMSA que vous utilisez dans les tâches.

b. Créez un secret dans AWS Secrets Manager, après avoir créé l'utilisateur dans Active Directory. Pour plus d'informations, voir Création d'un AWS Secrets Manager secret.

c. Saisissez respectivement le nom d'utilisateur, le mot de passe et le domaine de l'utilisateur dans les paires clé-valeur JSON appelées `username`, `password` et `domainName`.

```
{"username":"username","password":"passw0rd", "domainName":"example.com"}
```

d. Ajoutez la configuration au fichier CredSpec pour le compte de service. La `HostAccountConfig` supplémentaire contient l'Amazon Resource Name (ARN) du secret dans Secrets Manager.

   Sous Windows, le `PluginGUID` doit correspondre au GUID indiqué dans l'exemple de code suivant. Sous Linux, le `PluginGUID` est ignoré. Remplacez `MySecret` par l'exemple avec l'Amazon Resource Name (ARN) de votre secret.

```
    "ActiveDirectoryConfig": {
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
```

e. La fonctionnalité gMSA sans domaine nécessite des autorisations supplémentaires dans le rôle d'exécution des tâches. Suivez l'étape (Facultatif) secret gMSA sans domaine.

2. Configurez des instances et installez le démon **`credentials-fetcher`**.

   Vous pouvez installer le `credentials-fetcher` daemon avec un script de données utilisateur dans votre modèle de EC2 lancement. Les exemples suivants illustrent deux types de données utilisateur, `cloud-config` YAML ou le script bash. Ces exemples concernent Amazon Linux 2023 (AL2023). Remplacez `MyCluster` par le nom du cluster Amazon ECS que vous souhaitez que ces instances rejoignent.

   - **`cloud-config`** YAML

```
Content-Type: text/cloud-config
package_reboot_if_required: true
packages:
  # prerequisites
  - dotnet
  - realmd
  - oddjob
  - oddjob-mkhomedir
  - sssd
  - adcli
  - krb5-workstation
  - samba-common-tools
  # https://github.com/aws/credentials-fetcher gMSA credentials management for
 containers
  - credentials-fetcher
write_files:
# configure the ECS Agent to join your cluster.
# replace MyCluster with the name of your cluster.
- path: /etc/ecs/ecs.config
  owner: root:root
  permissions: '0644'
  content: |
    ECS_CLUSTER=MyCluster
    ECS_GMSA_SUPPORTED=true
runcmd:
# start the credentials-fetcher daemon and if it succeeded, make it start after
 every reboot
- "systemctl start credentials-fetcher"
- "systemctl is-active credentials-fetcher && systemctl enable credentials-
fetcher"
```

- Script bash

  Si vous êtes plus à l'aise avec les scripts bash et que vous avez plusieurs variables à écrire dans `/etc/ecs/ecs.config`, utilisez le format `heredoc` suivant. Ce format écrit tout entre les lignes commençant par cat et `EOF` dans le fichier de configuration.

```
#!/usr/bin/env bash
set -euxo pipefail

# prerequisites
timeout 30 dnf install -y dotnet realmd oddjob oddjob-mkhomedir sssd adcli
 krb5-workstation samba-common-tools
# install https://github.com/aws/credentials-fetcher gMSA credentials
 management for containers
timeout 30 dnf install -y credentials-fetcher

# start credentials-fetcher
systemctl start credentials-fetcher
systemctl is-active credentials-fetcher && systemctl enable credentials-fetcher

cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_GMSA_SUPPORTED=true
EOF
```

Il existe des variables de configuration facultatives pour le démon `credentials-fetcher` que vous pouvez définir dans `/etc/ecs/ecs.config`. Nous vous recommandons de définir les variables dans les données utilisateur du bloc YAML ou `heredoc` de manière similaire aux exemples précédents. Cela permet d'éviter les problèmes de configuration partielle qui peuvent survenir lors de la modification d'un fichier à plusieurs reprises. Pour plus d'informations sur la configuration de l'agent ECS, consultez Amazon ECS Container Agent on GitHub.

- Vous pouvez éventuellement utiliser la variable `CREDENTIALS_FETCHER_HOST` si vous modifiez la configuration du démon `credentials-fetcher` pour déplacer le socket vers un autre emplacement.

Configuration des autorisations et des secrets

Effectuez les étapes suivantes une fois pour chaque application et chaque définition de tâche. Nous vous recommandons d'utiliser le principe de moindre privilège et d'affiner les autorisations utilisées dans la stratégie. Ainsi, chaque tâche ne peut lire que les secrets dont elle a besoin.

1. (Facultatif) secret gMSA sans domaine

   Si vous utilisez la méthode sans domaine dans laquelle l'instance n'est pas jointe au domaine, procédez comme suit.

   Vous devez ajouter les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche. Cela permet au démon `credentials-fetcher` d'accéder au secret Secrets Manager. Remplacez l'exemple `MySecret` par l'Amazon Resource Name (ARN) de votre secret dans la liste `Resource`.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                   "arn:aws:ssm:arn:aws: 111122223333:secret:MySecret"
            ]
        }
    ]
}
```

> **Note**
>
> Si vous utilisez votre propre clé KMS pour chiffrer votre secret, vous devez ajouter les autorisations nécessaires à ce rôle et ajouter ce rôle à la politique des AWS KMS clés.

2. Décidez si vous utilisez SSM Parameter Store ou S3 pour stocker le CredSpec.

   Amazon ECS permet de référencer le chemin du fichier dans le champ `credentialSpecs` d'une définition de tâche.

   Si vous joignez les instances à un seul domaine, utilisez le préfixe `credentialspec:` au début de l'ARN dans la chaîne. Si vous utilisez le gMSA sans domaine, utilisez `credentialspecdomainless:`.

   Pour en savoir plus sur CredSpec, consultez Fichier de spécification des informations d'identification.

   - Compartiment Amazon S3

     Ajoutez la spécification d'informations d'identification à un compartiment Amazon S3. Référencez ensuite l'Amazon Resource Name (ARN) du compartiment Amazon S3 dans le champ `credentialSpecs` de la définition de tâche.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/
${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

     Pour permettre à vos tâches d'accéder au compartiment S3, ajoutez les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche Amazon ECS.

     JSON

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/{object}"
            ]
        }
    ]
}
```

- Paramètre SSM Parameter Store

  Ajoutez la spécification d'informations d'identification à un paramètre de SSM Parameter Store. Référencez ensuite l'Amazon Resource Name (ARN) du paramètre de SSM Parameter Store dans le champ `credentialSpecs` de la définition de tâche.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:ssm:aws-
region:111122223333:parameter/parameter_name"
            ],
            ...
        }
    ],
    ...
}
```

Pour permettre à vos tâches d'accéder au paramètre de SSM Parameter Store, ajoutez les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche Amazon ECS.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
            "arn:aws:ssm:arn:aws::111122223333:parameter/parameter_name"
            ]
        }
    ]
}
```

Fichier de spécification des informations d'identification

Amazon ECS utilise un fichier de spécification des informations d'identification Active Directory (CredSpec). Ce fichier contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur Linux. Vous générez le CredSpec et le référencez dans le champ `credentialSpecs` de votre définition de tâche. Le fichier CredSpec ne contient aucun secret.

Voici un exemple de fichier CredSpec.

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "example.com",
```

```
        "DnsName": "example.com",
        "NetBiosName": "example"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "example.com"
            }
        ],
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
    }
}
```

Création d’un CredSpec

Vous créez un CredSpec en utilisant le module PowerShell CredSpec sur un ordinateur Windows joint au domaine. Suivez les étapes décrites dans Créer une spécification d'informations d'identification sur le site Web Microsoft Learn.

Utilisation gMSA pour les Linux conteneurs sur Fargate

Amazon ECS prend en charge l'authentification Active Directory pour les conteneurs Linux sur Fargate via un type spécial de compte de service appelé compte de service géré de groupe (). gMSA

Les applications réseau Linux, comme les applications .NET Core, peuvent faire appel à Active Directory pour faciliter la gestion de l'authentification et des autorisations entre les utilisateurs et les services. Vous pouvez utiliser cette fonctionnalité en concevant des applications qui s'intègrent à Active Directory et s'exécutent sur des serveurs joints à un domaine. Toutefois, comme les conteneurs Linux ne peuvent pas être joints à un domaine, vous devez configurer un conteneur Linux pour qu'il soit exécuté avec gMSA.

Considérations

Tenez compte des points suivants avant d'utiliser gMSA des Linux conteneurs sur Fargate :

- Vous devez exécuter la version 1.4 ou ultérieure de Platform.
- Vous aurez peut-être besoin d'un ordinateur Windows joint au domaine pour remplir les conditions requises. Par exemple, vous pourriez avoir besoin d'un ordinateur Windows joint au domaine pour créer le gMSA dans Active Directory avec PowerShell. Les PowerShell outils RSAT Active Director ne sont disponibles que pourWindows. Pour plus d'informations, veuillez consulter Installation des outils d'administration Active Directory (langue française non garantie).
- Vous devez utiliser le mode sans domaine. gMSA

  Amazon ECS utilise un fichier de spécification des informations d'identification Active Directory (CredSpec). Ce fichier contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur. Vous générez le CredSpec fichier, puis vous le stockez dans un compartiment Amazon S3.
- Une tâche ne peut prendre en charge qu'un seul Active Directory.

Prérequis

Avant d'utiliser la fonctionnalité gMSA pour les conteneurs Linux avec Amazon ECS, assurez-vous de remplir les conditions suivantes :

- Vous configurez un domaine Active Directory avec les ressources auxquelles vous souhaitez que vos conteneurs accèdent. Amazon ECS prend en charge les configurations suivantes :
  - Un AWS Directory Service Active Directory. AWS Directory Service est un Active Directory AWS géré hébergé sur Amazon EC2. Pour plus d'informations, consultez Getting Started with AWS Managed Microsoft AD dans le Guide d'AWS Directory Service administration.
  - Un répertoire Active Directory sur site. Vous devez vous assurer que l'instance de conteneur Linux Amazon ECS peut se joindre au domaine. Pour de plus amples informations, veuillez consulter AWS Direct Connect.
- Vous disposez d'un gMSA compte existant dans Active Directory et d'un utilisateur autorisé à accéder au compte gMSA de service. Pour de plus amples informations, veuillez consulter Création d'un utilisateur Active Directory pour gMSA sans domaine.
- Vous disposez d'un compartiment Amazon S3. Pour plus d'informations, consultez la section Création d'un compartiment dans le guide de l'utilisateur Amazon S3.

Configuration de conteneurs Linux compatibles avec gMSA sur Amazon ECS

Préparation de l'infrastructure

Les étapes suivantes sont des considérations et une configuration qui ne sont effectuées qu'une seule fois.

- Création d'un utilisateur Active Directory pour gMSA sans domaine

  Lorsque vous utilisez l'option sans domainegMSA, le conteneur n'est pas joint au domaine. Les autres applications qui s'exécutent sur le conteneur ne peuvent pas utiliser les informations d'identification pour accéder au domaine. Les tâches qui utilisent un domaine différent peuvent être exécutées sur le même conteneur. Vous indiquez le nom d'un secret AWS Secrets Manager dans le CredSpec fichier. Le secret doit contenir un nom d'utilisateur, un mot de passe et le domaine auquel se connecter.

  Cette fonctionnalité est similaire à la fonctionnalité gMSA support for non-domain-joined container hosts. Pour plus d'informations sur la fonctionnalité Windows, veuillez consulter Architecture et améliorations des gMSA sur le site Web de Microsoft Learn.

  a. Configurez un utilisateur dans votre domaine Active Directory. L'utilisateur d'Active Directory doit être autorisé à accéder au compte gMSA de service que vous utilisez pour les tâches.
  b. Vous disposez d'un VPC et de sous-réseaux capables de résoudre le nom de domaine Active Directory. Configurez le VPC avec les options DHCP avec le nom de domaine qui pointe vers le nom du service Active Directory. Pour plus d'informations sur la configuration des options DHCP pour un VPC, consultez la section Travailler avec des ensembles d'options DHCP dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.
  c. Créez un secret dans AWS Secrets Manager.
  d. Créez le fichier de spécification des informations d'identification.

Configuration des autorisations et des secrets

Effectuez les étapes suivantes une fois pour chaque application et chaque définition de tâche. Nous vous recommandons d'utiliser le principe de moindre privilège et d'affiner les autorisations utilisées dans la stratégie. Ainsi, chaque tâche ne peut lire que les secrets dont elle a besoin.

1. Créez un utilisateur dans votre domaine Active Directory. L'utilisateur d'Active Directory doit être autorisé à accéder aux comptes de service gMSA que vous utilisez dans les tâches.
2. Après avoir créé l'utilisateur Active Directory, créez un secret dans AWS Secrets Manager. Pour plus d'informations, veuillez consulter Créer un secret AWS Secrets Manager.
3. Saisissez respectivement le nom d'utilisateur, le mot de passe et le domaine de l'utilisateur dans les paires clé-valeur JSON appelées `username`, `password` et `domainName`.

```
{"username":"username","password":"passw0rd", "domainName":"example.com"}
```

4. Vous devez ajouter les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche. Cela permet au démon `credentials-fetcher` d'accéder au secret Secrets Manager. Remplacez l'exemple `MySecret` par l'Amazon Resource Name (ARN) de votre secret dans la liste `Resource`.

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
            "arn:aws:secretsmanager:arn:aws::111122223333:secret:MySecret"
            ]
        }
    ]
}
```

> **Note**
>
> Si vous utilisez votre propre clé KMS pour chiffrer votre secret, vous devez ajouter les autorisations nécessaires à ce rôle et ajouter ce rôle à la politique des AWS KMS clés.

5. Ajoutez la spécification d'informations d'identification à un compartiment Amazon S3. Référencez ensuite l'Amazon Resource Name (ARN) du compartiment Amazon S3 dans le champ `credentialSpecs` de la définition de tâche.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

Pour permettre à vos tâches d'accéder au compartiment S3, ajoutez les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche Amazon ECS.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListObjects"
            ],
            "Resource": [
                "arn:aws:s3:::{bucket_name}",
                "arn:aws:s3:::{bucket_name}/{object}"
            ]
        }
    ]
```

```
}
```

Fichier de spécification des informations d'identification

Amazon ECS utilise un fichier de spécification des informations d'identification Active Directory (CredSpec). Ce fichier contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur Linux. Vous générez le CredSpec et le référencez dans le champ `credentialSpecs` de votre définition de tâche. Le fichier CredSpec ne contient aucun secret.

Voici un exemple de fichier CredSpec.

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "example.com",
        "DnsName": "example.com",
        "NetBiosName": "example"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "example.com"
            }
        ],
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
    }
}
```

Création d'un CredSpec fichier et chargement de celui-ci sur un Amazon S3

Vous créez un CredSpec en utilisant le module PowerShell CredSpec sur un ordinateur Windows joint au domaine. Suivez les étapes décrites dans Créer une spécification d'informations d'identification sur le site Web Microsoft Learn.

Après avoir créé le fichier de spécification des informations d'identification, chargez-le dans un compartiment Amazon S3. Copiez le fichier CredSpec sur l'ordinateur ou l'environnement dans lequel vous exécutez les commandes AWS CLI .

Exécutez la AWS CLI commande suivante pour le CredSpec télécharger sur Amazon S3. Remplacez `amzn-s3-demo-bucket` par le nom de votre compartiment Simple Storage Service (Amazon S3). Vous pouvez stocker le fichier sous forme d'objet dans n'importe quel compartiment et à n'importe quel emplacement, mais vous devez autoriser l'accès à ce compartiment et à cet emplacement dans la stratégie que vous associez au rôle d'exécution des tâches.

Pour PowerShell, utilisez la commande suivante :

```
$ Write-S3Object -BucketName "amzn-s3-demo-bucket" -Key "ecs-domainless-gmsa-credspec"
 -File "gmsa-cred-spec.json"
```

La AWS CLI commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` les interpréteurs de commandes compatibles.

```
$ aws s3 cp gmsa-cred-spec.json \
s3://amzn-s3-demo-bucket/ecs-domainless-gmsa-credspec
```

Utilisation de conteneurs Windows Amazon ECS avec des conteneurs sans domaine à gMSA l'aide du AWS CLI

Le didacticiel suivant explique comment créer une tâche Amazon ECS qui exécute un conteneur Windows qui possède des informations d'identification pour accéder à Active Directory avec l' AWS CLI. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.

Rubriques

- Prérequis

- Étape 1 : créer et configurer le compte gMSA sur les services de domaine Active Directory (AD DS)
- Étape 2 : charger les informations d'identification dans Secrets Manager
- Étape 3 : modifier votre JSON CredSpec pour inclure des informations gMSA sans domaine
- Étape 4 : charger CredSpec dans Amazon S3
- Étape 5 (facultative) : créer un cluster Amazon ECS
- Étape 6 : créer un rôle IAM pour les instances de conteneur
- Étape 7 : créer un rôle d'exécution de tâche personnalisé
- Étape 8 : créer un rôle de tâche pour Amazon ECS Exec
- Étape 9 : enregistrer une définition de tâche qui utilise gMSA sans domaine
- Étape 10 : enregistrer une instance de conteneur Windows dans le cluster
- Étape 11 : vérifier l'instance de conteneur
- Étape 12 : exécuter une tâche Windows
- Étape 13 : vérifier que le conteneur possède les informations d'identification gMSA
- Étape 14 : nettoyer
- Débogage des gMSA Amazon ECS sans domaine pour les conteneurs Windows

Prérequis

Le didacticiel suppose de remplir les prérequis suivants :

- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique Amazon ECS_ FullAccess IAM.
- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations sur l'installation ou la mise à niveau de votre AWS CLI AWS Command Line Interface.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir de AWS CLI SDKs, et de l'API Amazon ECS via les deux IPv4 et. IPv6 Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

- Vous configurez un domaine Active Directory avec les ressources auxquelles vous souhaitez que vos conteneurs accèdent. Amazon ECS prend en charge les configurations suivantes :
  - Un AWS Directory Service Active Directory. AWS Directory Service est un Active Directory AWS géré hébergé sur Amazon EC2. Pour plus d'informations, consultez Getting Started with AWS Managed Microsoft AD dans le Guide d'AWS Directory Service administration.
  - Un répertoire Active Directory sur site. Vous devez vous assurer que l'instance de conteneur Linux Amazon ECS peut se joindre au domaine. Pour de plus amples informations, veuillez consulter AWS Direct Connect.
- Vous disposez d'un VPC et de sous-réseaux capables de résoudre le nom de domaine Active Directory.
- Vous avez choisi entre les gMSA sans domaine et joindre chaque instance à un seul domaine. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.

  Choisissez ensuite le stockage de données pour CredSpec et, éventuellement, pour les informations d'identification utilisateur Active Directory pour les gMSA sans domaine.

  Amazon ECS utilise un fichier de spécification des informations d'identification Active Directory (CredSpec). Ce fichier contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur. Vous générez le fichier CredSpec, puis vous le stockez dans l'une des options de stockage CredSpec du tableau suivant, spécifique au système d'exploitation des instances de conteneur. Pour utiliser la méthode sans domaine, une section facultative du fichier CredSpec peut spécifier les informations d'identification dans l'une des options de stockage domainless user credentials du tableau suivant, spécifiques au système d'exploitation des instances de conteneur.

Emplacement de stockage	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	informations d'identification utilisateur sans domaine	informations d'identification utilisateur sans domaine

Emplacement de stockage	Linux	Windows
Boutique de paramètres Amazon EC2 Systems Manager	CredSpec	CredSpec, informations d'identification utilisateur sans domaine
Fichier local	N/A	CredSpec

- (Facultatif) AWS CloudShell est un outil qui fournit aux clients une ligne de commande sans avoir à créer leur propre EC2 instance. Pour plus d'informations, voir Qu'est-ce que c'est AWS CloudShell ? dans le guide de AWS CloudShell l'utilisateur.

Étape 1 : créer et configurer le compte gMSA sur les services de domaine Active Directory (AD DS)

Créez et configurez un compte gMSA sur le domaine Active Directory.

> **Note**
>
> Cette étape crée deux comptes distincts : un compte de service géré par le groupe (GMSA) qui fournit l'identité de vos conteneurs, et un compte utilisateur normal utilisé pour l'authentification de domaine. Ces comptes ont des objectifs différents et doivent porter des noms différents.

1. Génération d'une clé racine du service de diffusion de clés

   > **Note**
   >
   > Si vous utilisez AWS Directory Service, vous pouvez ignorer cette étape.

   La clé racine KDS et les autorisations gMSA sont configurées avec votre AD Microsoft géré par AWS .

   Si vous n'avez pas encore créé de compte de service gMSA dans votre domaine, vous devez d'abord générer une clé racine du service de diffusion de clés (KDS). Le KDS est responsable de la création, de la rotation et de la diffusion du mot de passe gMSA aux hôtes autorisés. Lorsque

`ccg.exe` a besoin de récupérer les informations d'identification gMSA, il contacte KDS pour récupérer le mot de passe actuel.

Pour vérifier si la clé racine KDS a déjà été créée, exécutez l' PowerShellapplet de commande suivante avec les privilèges d'administrateur de domaine sur un contrôleur de domaine à l'aide du module. `ActiveDirectory` PowerShell Pour plus d'informations sur le module, voir ActiveDirectory Module sur le site Web de Microsoft Learn.

```
PS C:\> Get-KdsRootKey
```

Si la commande renvoie un ID de clé, vous pouvez ignorer le reste de cette étape. Dans le cas contraire, créez la clé racine KDS en exécutant la commande suivante :

```
PS C:\> Add-KdsRootKey -EffectiveImmediately
```

Bien que l'argument `EffectiveImmediately` de la commande implique que la clé est effective immédiatement, vous devez attendre 10 heures avant que la clé racine KDS soit répliquée et puisse être utilisée sur tous les contrôleurs de domaine.

2. Création du compte gMSA

   Pour créer le gMSA compte et autoriser le `ccg.exe` à récupérer le gMSA mot de passe, exécutez les PowerShell commandes suivantes depuis un serveur ou un client Windows ayant accès au domaine. Remplacez-le `ExampleAccount` par le nom que vous souhaitez pour votre gMSA compte et remplacez-le `example-domain` par votre nom de domaine Active Directory (par exemple, si votre domaine est`contoso.com`, utilisez`contoso`).

   a.
   ```
   PS C:\> Install-WindowsFeature RSAT-AD-PowerShell
   ```

   b.
   ```
   PS C:\> New-ADGroup -Name "ExampleAccount Authorized Hosts" -SamAccountName
    "ExampleAccountHosts" -GroupScope DomainLocal
   ```

   c.
   ```
   PS C:\> New-ADServiceAccount -Name "ExampleAccount" -DnsHostName "example-
   domain" -ServicePrincipalNames "host/ExampleAccount", "host/example-domain" -
   PrincipalsAllowedToRetrieveManagedPassword "ExampleAccountHosts"
   ```

   d. Créez un utilisateur avec un mot de passe permanent qui n'expire pas. Ces informations d'identification sont stockées AWS Secrets Manager et utilisées par chaque tâche pour rejoindre le domaine. Il s'agit d'un compte utilisateur distinct du compte GMSA créé ci-

dessus. `ExampleServiceUser`Remplacez-le par le nom que vous souhaitez pour ce compte utilisateur de service.

```
PS C:\> New-ADUser -Name "ExampleServiceUser" -AccountPassword (ConvertTo-
SecureString -AsPlainText "Test123" -Force) -Enabled 1 -PasswordNeverExpires 1
```

e.
```
PS C:\> Add-ADGroupMember -Identity "ExampleAccountHosts" -Members
 "ExampleServiceUser"
```

f. Installez le PowerShell module de création d'CredSpecobjets dans Active Directory et générez le CredSpec JSON.

```
PS C:\> Install-PackageProvider -Name NuGet -Force
```

```
PS C:\> Install-Module CredentialSpec
```

g.
```
PS C:\> New-CredentialSpec -AccountName ExampleAccount
```

3. Copiez le résultat JSON de la commande précédente dans un fichier appelé `gmsa-cred-spec.json`. Il s'agit du fichier CredSpec. Il est utilisé à l'étape 3, Étape 3 : modifier votre JSON CredSpec pour inclure des informations gMSA sans domaine.

Étape 2 : charger les informations d'identification dans Secrets Manager

Copiez les informations d'identification Active Directory dans un système de stockage d'informations d'identification sécurisé, afin que chaque tâche les récupère. Il s'agit de la méthode gMSA sans domaine. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

- Exécutez la AWS CLI commande suivante et remplacez le nom d'utilisateur, le mot de passe et le nom de domaine en fonction de votre environnement. Utilisez le nom du compte utilisateur du service (et non le nom du compte GMSA) comme nom d'utilisateur. Conservez l'ARN du

secret pour l'utiliser à l'étape suivante, Étape 3 : modifier votre JSON CredSpec pour inclure des informations gMSA sans domaine.

La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

```
$ aws secretsmanager create-secret \
--name gmsa-plugin-input \
--description "Amazon ECS - gMSA Portable Identity." \
--secret-string "{\"username\":\"ExampleServiceUser\",\"password\":\"Test123\",
\"domainName\":\"contoso.com\"}"
```

Étape 3 : modifier votre JSON CredSpec pour inclure des informations gMSA sans domaine

Avant de charger le CredSpec vers l'une des options de stockage, ajoutez des informations au CredSpec à l'aide de l'ARN du secret dans Secrets Manager provenant de l'étape précédente. Pour plus d'informations, consultez la section Configuration des spécifications d'identification supplémentaires pour le cas d'utilisation d' non-domain-joinedun hôte de conteneur sur le site Web de Microsoft Learn.

1. Ajoutez les informations suivantes au fichier CredSpec contenu dans le `ActiveDirectoryConfig`. Remplacez l'ARN par le secret de l'étape précédente dans Secrets Manager.

   Veuillez noter que la valeur `PluginGUID` doit correspondre au GUID de l'extrait de code suivant et qu'elle est obligatoire.

```
"HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
```

Vous pouvez également utiliser un secret dans SSM Parameter Store en utilisant l'ARN au format suivant : `\"arn:aws:ssm:aws-region:111122223333:parameter/gmsa-plugin-input\"`.

2. Une fois le fichier CredSpec modifié, il doit ressembler à l'exemple suivant :

```
{
  "CmsPlugins": [
    "ActiveDirectory"
  ],
  "DomainJoinConfig": {
    "Sid": "S-1-5-21-4066351383-705263209-1606769140",
    "MachineAccountName": "ExampleAccount",
    "Guid": "ac822f13-583e-49f7-aa7b-284f9a8c97b6",
    "DnsTreeName": "example-domain",
    "DnsName": "example-domain",
    "NetBiosName": "example-domain"
  },
  "ActiveDirectoryConfig": {
    "GroupManagedServiceAccounts": [
      {
        "Name": "ExampleAccount",
        "Scope": "example-domain"
      },
      {
        "Name": "ExampleAccount",
        "Scope": "example-domain"
      }
    ],
    "HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
  }
}
```

Étape 4 : charger CredSpec dans Amazon S3

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

1. Copiez le fichier CredSpec sur l'ordinateur ou l'environnement dans lequel vous exécutez les commandes AWS CLI .
2. Exécutez la AWS CLI commande suivante pour le CredSpec télécharger sur Amazon S3. Remplacez `MyBucket` par le nom de votre compartiment Simple Storage Service (Amazon S3). Vous pouvez stocker le fichier sous forme d'objet dans n'importe quel compartiment et à n'importe quel emplacement, mais vous devez autoriser l'accès à ce compartiment et à cet emplacement dans la stratégie que vous associez au rôle d'exécution des tâches.

   La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

   ```
   $ aws s3 cp gmsa-cred-spec.json \
   s3://MyBucket/ecs-domainless-gmsa-credspec
   ```

Étape 5 (facultative) : créer un cluster Amazon ECS

Par défaut, votre compte possède un cluster Amazon ECS nommé `default`. Ce cluster est utilisé par défaut dans AWS CloudFormation les AWS CLI SDKs Vous pouvez utiliser des clusters supplémentaires pour regrouper et organiser les tâches et l'infrastructure, et attribuer des valeurs par défaut pour certaines configurations.

Vous pouvez créer un cluster à partir du AWS Management Console AWS CLI, SDKs, ou AWS CloudFormation. Les paramètres et la configuration du cluster n'ont aucune incidence sur gMSA.

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

```
$ aws ecs create-cluster --cluster-name windows-domainless-gmsa-cluster
```

> **Important**
>
> Si vous choisissez de créer votre propre cluster, vous devez spécifier `--cluster clusterName` pour chaque commande que vous prévoyez d'utiliser avec ce cluster.

Étape 6 : créer un rôle IAM pour les instances de conteneur

Une instance de conteneur est un ordinateur hôte permettant d'exécuter des conteneurs dans des tâches ECS, par exemple EC2 des instances Amazon. Chaque instance de conteneur s'enregistre dans un cluster Amazon ECS. Avant de lancer EC2 des instances Amazon et de les enregistrer dans un cluster, vous devez créer un rôle IAM que vos instances de conteneur pourront utiliser.

Pour créer le rôle d'instance de conteneur, veuillez consulter Rôle IAM d'instance de conteneur Amazon ECS. Le `ecsInstanceRole` par défaut dispose des autorisations suffisantes pour terminer ce didacticiel.

Étape 7 : créer un rôle d'exécution de tâche personnalisé

Amazon ECS peut utiliser un rôle IAM différent pour les autorisations nécessaires au démarrage de chaque tâche, au lieu du rôle d'instance de conteneur. Il s'agit du rôle d'exécution de tâche. Nous recommandons de créer un rôle d'exécution de tâche avec uniquement les autorisations requises pour qu'ECS exécute la tâche, également appelées autorisations de moindre privilège. Pour plus d'informations sur le principe du moindre privilège, veuillez consulter SEC03-BP02 Accorder un accès selon le principe du moindre privilège dans le cadre AWS Well-Architected.

1. Pour créer un rôle d'exécution de tâche, veuillez consulter Création du rôle d'exécution de tâche. Les autorisations par défaut permettent à l'instance de conteneur d'extraire des images de conteneur d'Amazon Elastic Container Registry `stdout` et `stderr` de vos applications pour les connecter à Amazon CloudWatch Logs.

   Comme le rôle nécessite des autorisations personnalisées pour ce didacticiel, vous pouvez lui donner un nom différent de `ecsTaskExecutionRole`. Ce didacticiel utilise `ecsTaskExecutionRole` dans les étapes suivantes.

2. Ajoutez les autorisations suivantes en créant une stratégie personnalisée, soit une stratégie en ligne qui n'existe que pour ce rôle, soit une stratégie que vous pouvez réutiliser. Remplacez l'ARN de la `Resource` dans la première instruction par le compartiment et l'emplacement Amazon S3, et la seconde `Resource` par l'ARN du secret dans Secrets Manager.

   Si vous chiffrez le secret dans Secrets Manager avec une clé personnalisée, vous devez également autoriser `kms:Decrypt` pour la clé.

   Si vous utilisez SSM Parameter Store au lieu de Secrets Manager, vous devez autoriser `ssm:GetParameter` pour le paramètre, au lieu de `secretsmanager:GetSecretValue`.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::MyBucket/ecs-domainless-gmsa-credspec/
gmsa-cred-spec.json"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource":
 "arn:aws:secretsmanager:arn:aws:: 111122223333:secret:gmsa-plugin-AbCdEf"
        }
    ]
}
```

Étape 8 : créer un rôle de tâche pour Amazon ECS Exec

Ce didacticiel utilise Amazon ECS Exec pour vérifier le fonctionnement en exécutant une commande dans une tâche en cours d'exécution. Pour utiliser ECS Exec, le service ou la tâche doit activer ECS Exec, et le rôle de tâche (mais pas le rôle d'exécution de tâche) doit disposer d'autorisations `ssmmessages`. Pour la politique IAM requise, veuillez consulter Autorisations ECS Exec.

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

Pour créer un rôle de tâche à l'aide de AWS CLI, procédez comme suit.

1. Créez un fichier nommé `ecs-tasks-trust-policy.json` avec le contenu suivant :

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

2. Créez un rôle IAM. Vous pouvez remplacer le nom `ecs-exec-demo-task-role` mais le conserver pour les étapes suivantes.

   La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

```
$ aws iam create-role --role-name ecs-exec-demo-task-role \
--assume-role-policy-document file://ecs-tasks-trust-policy.json
```

   Vous pouvez supprimer le fichier `ecs-tasks-trust-policy.json`.

3. Créez un fichier nommé `ecs-exec-demo-task-role-policy.json` avec le contenu suivant :

   JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssmmessages:CreateControlChannel",
                "ssmmessages:CreateDataChannel",
```

```
                "ssmmessages:OpenControlChannel",
                "ssmmessages:OpenDataChannel"
            ],
            "Resource": "*"
        }
    ]
}
```

4. Créez une politique IAM et attachez-la au rôle de l'étape précédente.

   La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

```
$ aws iam put-role-policy \
    --role-name ecs-exec-demo-task-role \
    --policy-name ecs-exec-demo-task-role-policy \
    --policy-document file://ecs-exec-demo-task-role-policy.json
```

   Vous pouvez supprimer le fichier `ecs-exec-demo-task-role-policy.json`.

Étape 9 : enregistrer une définition de tâche qui utilise gMSA sans domaine

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

1. Créez un fichier nommé `windows-gmsa-domainless-task-def.json` avec le contenu suivant :

```
{
  "family": "windows-gmsa-domainless-task",
  "containerDefinitions": [
    {
      "name": "windows_sample_app",
      "image": "mcr.microsoft.com/windows/servercore/iis",
      "cpu": 1024,
      "memory": 1024,
      "essential": true,
      "credentialSpecs": [
```

```
                "credentialspecdomainless:arn:aws:s3:::ecs-domainless-gmsa-
credspec/gmsa-cred-spec.json"
      ],
      "entryPoint": [
        "powershell",
        "-Command"
      ],
      "command": [
        "New-Item -Path C:\\inetpub\\wwwroot\\index.html -ItemType file -Value
 '<html> <head> <title>Amazon ECS Sample App</title> <style>body {margin-top:
 40px; background-color: #333;} </style> </head><body> <div style=color:white;text-
align:center> <h1>Amazon ECS Sample App</h1> <h2>Congratulations!</h2> <p>Your
 application is now running on a container in Amazon ECS.</p>' -Force ; C:\
\ServiceMonitor.exe w3svc"
      ],
      "portMappings": [
        {
          "protocol": "tcp",
          "containerPort": 80,
          "hostPort": 8080
        }
      ]
    }
  ],
  "taskRoleArn": "arn:aws:iam::111122223333:role/ecs-exec-demo-task-role",
  "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
}
```

2. Enregistrez la définition de tâche en exécutant la commande suivante :

   La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

```
$ aws ecs register-task-definition \
--cli-input-json file://windows-gmsa-domainless-task-def.json
```

Étape 10 : enregistrer une instance de conteneur Windows dans le cluster

Lancez une instance Amazon EC2 Windows et exécutez l'agent de conteneur ECS pour l'enregistrer en tant qu'instance de conteneur dans le cluster. ECS exécute des tâches sur les instances de conteneur enregistrées dans le cluster dans lequel les tâches sont démarrées.

1. Pour lancer une instance Amazon EC2 Windows configurée pour Amazon ECS dans le AWS Management Console, consultezLancement d'une instance de conteneur Amazon ECS Windows. Arrêtez-vous à l'étape des données utilisateur.
2. Pour gMSA, les données utilisateur doivent définir la variable d'environnement `ECS_GMSA_SUPPORTED` avant de démarrer l'agent de conteneur ECS.

   Pour ECS Exec, l'agent doit commencer par l'argument `-EnableTaskIAMRole`.

   Pour sécuriser le rôle IAM de l'instance en empêchant les tâches d'atteindre le service Web EC2 IMDS pour récupérer les informations d'identification du rôle, ajoutez l'argument. `-AwsvpcBlockIMDS` Cela s'applique uniquement aux tâches qui utilisent le mode réseau `awsvpc`.

   ```
   <powershell>
   [Environment]::SetEnvironmentVariable("ECS_GMSA_SUPPORTED", $TRUE, "Machine")
   Import-Module ECSTools
   Initialize-ECSAgent -Cluster windows-domainless-gmsa-cluster -EnableTaskIAMRole -
   AwsvpcBlockIMDS
   </powershell>
   ```
3. Consultez un résumé de la configuration de votre instance dans le panneau Summary (Récapitulatif) et, lorsque vous êtes prêt, choisissez Launch instance (Lancer l’instance).

Étape 11 : vérifier l'instance de conteneur

Vous pouvez vérifier qu'il existe une instance de conteneur dans le cluster à l'aide de l' AWS Management Console. Cependant, gMSA nécessite des fonctionnalités supplémentaires indiquées sous forme d'attributs. Ces attributs ne sont pas visibles dans le AWS Management Console. Ce didacticiel utilise donc le AWS CLI.

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

1. Répertoriez les instances de conteneur dans le cluster. Les instances de conteneur ont un identifiant différent de celui de l' EC2 instance.

   ```
   $ aws ecs list-container-instances
   ```

   Sortie :

   ```
   {
       "containerInstanceArns": [
           "arn:aws:ecs:aws-region:111122223333:container-
   instance/default/MyContainerInstanceID"
       ]
   }
   ```

   Par exemple, `526bd5d0ced448a788768334e79010fd` est un ID d'instance de conteneur valide.

2. Utilisez l'ID d'instance de conteneur indiqué à l'étape précédente pour obtenir les détails de l'instance de conteneur. Remplacez `MyContainerInstanceID` par l'ID.

   La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

   ```
   $ aws ecs describe-container-instances \
        ----container-instances MyContainerInstanceID
   ```

   Veuillez noter que la sortie est très longue.

3. Vérifiez que la liste `attributes` contient un objet avec la clé appelée `name` et une valeur `ecs.capability.gmsa-domainless`. Voici un exemple de l'objet.

   Sortie :

   ```
   {
       "name": "ecs.capability.gmsa-domainless"
   }
   ```

Étape 12 : exécuter une tâche Windows

Exécutez une tâche Amazon ECS. S'il n'y a qu'une seule instance de conteneur dans le cluster, vous pouvez utiliser `run-task`. S'il existe de nombreuses instances de conteneur différentes, il peut être plus facile d'utiliser `start-task` et de spécifier l'ID d'instance de conteneur sur lequel exécuter la tâche, plutôt que d'ajouter des contraintes de placement à la définition de la tâche pour contrôler le type d'instance de conteneur sur lequel exécuter cette tâche.

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

1. La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

   ```
   aws ecs run-task --task-definition windows-gmsa-domainless-task \
       --enable-execute-command --cluster windows-domainless-gmsa-cluster
   ```

   Notez l'ID de tâche qui est renvoyé par la commande.
2. Exécutez la commande suivante pour vérifier que la tâche a démarré. Cette commande attend et ne renvoie pas l'invite du shell tant que la tâche ne démarre pas. Remplacez `MyTaskID` par l'ID de tâche de l'étape précédente.

   ```
   $ aws ecs wait tasks-running --task MyTaskID
   ```

Étape 13 : vérifier que le conteneur possède les informations d'identification gMSA

Vérifiez que le conteneur de la tâche possède un jeton Kerberos. gMSA

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

1. La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

```
$ aws ecs execute-command \
--task MyTaskID \
--container windows_sample_app \
--interactive \
--command powershell.exe
```

Le résultat sera une PowerShell invite.

2. Exécutez la commande suivante dans le PowerShell terminal à l'intérieur du conteneur.

```
PS C:\> klist get ExampleAccount$
```

Dans la sortie, notez que le `Principal` est celui que vous avez créé précédemment.

Étape 14 : nettoyer

Une fois que vous avez terminé ce didacticiel, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources inutilisées.

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

1. Arrêtez la tâche. Remplacez `MyTaskID` par l'ID de tâche de l'étape 12, Étape 12 : exécuter une tâche Windows.

```
$ aws ecs stop-task --task MyTaskID
```

2. Mettez fin à l' EC2 instance Amazon. Ensuite, l'instance de conteneur du cluster sera automatiquement supprimée au bout d'une heure.

   Vous pouvez rechercher et résilier l'instance à l'aide de la EC2 console Amazon. Vous pouvez également exécuter la commande suivante. Pour exécuter la commande, recherchez l'ID d' EC2 instance dans le résultat de la `aws ecs describe-container-instances` commande de l'étape 1Étape 11 : vérifier l'instance de conteneur. i-10a64379 est un exemple d'ID d'instance. EC2

```
$ aws ec2 terminate-instances --instance-ids MyInstanceID
```

3. Supprimez le fichier CredSpec dans Amazon S3. Remplacez `MyBucket` par le nom de votre compartiment Simple Storage Service (Amazon S3).

```
$ aws s3api delete-object --bucket MyBucket --key ecs-domainless-gmsa-credspec/
gmsa-cred-spec.json
```

4. Supprimez le secret dans Secrets Manager. Si vous avez plutôt utilisé SSM Parameter Store, supprimez le paramètre.

   La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

```
$ aws secretsmanager delete-secret --secret-id gmsa-plugin-input \
     --force-delete-without-recovery
```

5. Annulez l'enregistrement et supprimez la définition de tâche. En annulant l'enregistrement de la définition de tâche, vous la marquez comme inactive afin qu'elle ne puisse pas être utilisée pour démarrer de nouvelles tâches. Ensuite, vous pouvez supprimer la définition de tâche.

   a. Annulez l'enregistrement de la définition de tâche en spécifiant la version. ECS crée automatiquement des versions des définitions de tâches, qui sont numérotées à partir de 1. Vous faites référence aux versions dans le même format que les étiquettes sur les images des conteneurs, telles que `:1`.

```
$ aws ecs deregister-task-definition --task-definition windows-gmsa-domainless-
task:1
```

   b. Supprimez la définition de tâche.

```
$ aws ecs delete-task-definitions --task-definition windows-gmsa-domainless-
task:1
```

6. (Facultatif) Supprimez le cluster ECS, si vous en avez créé un.

```
$ aws ecs delete-cluster --cluster windows-domainless-gmsa-cluster
```

Débogage des gMSA Amazon ECS sans domaine pour les conteneurs Windows

Statut de tâche Amazon ECS

ECS essaie de démarrer une tâche une seule fois. Toute tâche présentant un problème est arrêtée et définie sur le statut `STOPPED`. Il existe deux types courants de problèmes liés aux tâches. Premièrement, les tâches qui n'ont pas pu être démarrées. Deuxièmement, les tâches pour lesquelles l'application s'est arrêtée dans l'un des conteneurs. Dans le AWS Management Console champ Motif de l'arrêt de la tâche, recherchez la raison pour laquelle la tâche a été arrêtée. Dans l' AWS CLI, décrivez la tâche et examinez la `stoppedReason`. Pour les étapes à suivre dans AWS Management Console le AWS CLI sable, voirL'affichage d'Amazon ECS a permis de stopper les erreurs de tâches.

Événements Windows

Les événements Windows pour gMSA dans les conteneurs sont enregistrés dans le fichier journal `Microsoft-Windows-Containers-CCG` et se trouvent dans l'Observateur d'événements, dans la section Applications et services de `Logs\Microsoft\Windows\Containers-CCG\Admin`. Pour obtenir d'autres conseils de débogage, consultez la section Résoudre les problèmes liés aux conteneurs g MSAs pour Windows sur le site Web de Microsoft Learn.

Plug-in gMSA d'agent ECS

La journalisation du plug-in gMSA pour l'agent ECS sur l'instance de conteneur Windows se trouve dans le répertoire suivant, `C:/ProgramData/Amazon/gmsa-plugin/`. Consultez ce journal pour voir si les informations d'identification de l'utilisateur sans domaine ont été téléchargées depuis l'emplacement de stockage, tel que Secrets Manager, et si le format des informations d'identification a été lu correctement.

Découvrez comment utiliser les conteneurs g MSAs pour EC2 Windows pour Amazon ECS

Amazon ECS prend en charge l'authentification Active Directory pour les conteneurs Windows par l'intermédiaire d'un type de compte de service spécial appelé gMSA (group Managed Service Account).

Les applications réseau Windows comme les applications .NET font souvent appel à Active Directory pour faciliter la gestion de l'authentification et des autorisations entre les utilisateurs et les services. Les développeurs conçoivent généralement leurs applications de façon qu'elles s'intègrent à Active Directory et s'exécutent à cette fin sur des serveurs joints à un domaine. Étant donné que les conteneurs Windows ne peuvent pas être joints à un domaine, vous devez configurer un conteneur Windows pour qu'il s'exécute avec un compte gMSA.

Un conteneur Windows exécuté avec GMSA s'appuie sur son EC2 instance Amazon hôte pour récupérer les informations d'identification GMSA auprès du contrôleur de domaine Active Directory et les fournir à l'instance de conteneur. Pour plus d'informations, voir Create g MSAs pour les conteneurs Windows.

> **Note**
>
> Cette fonction n'est pas prise en charge par les conteneurs Windows sur Fargate.

Rubriques

- Considérations
- Prérequis
- Configuration de gMSA pour des conteneurs Windows sur Amazon ECS

Considérations

Les points suivants doivent être pris en compte lors de l'utilisation de g MSAs pour les conteneurs Windows :

- Lorsque vous utilisez l'AMI Windows Server 2016 Full optimisée pour Amazon ECS pour vos instances de conteneur, le nom d'hôte du conteneur doit être le même que le nom de compte gMSA défini dans le fichier de spécification des informations d'identification. Pour spécifier le nom d'hôte d'un conteneur, utilisez le paramètre de définition de conteneur `hostname`. Pour de plus amples informations, veuillez consulter Paramètres réseau.
- Vous avez choisi entre les gMSA sans domaine et joindre chaque instance à un seul domaine. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.

Choisissez ensuite le stockage de données pour CredSpec et, éventuellement, pour les informations d'identification utilisateur Active Directory pour les gMSA sans domaine.

Amazon ECS utilise un fichier de spécification des informations d'identification Active Directory (CredSpec). Ce fichier contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur. Vous générez le fichier CredSpec, puis vous le stockez dans l'une des options de stockage CredSpec du tableau suivant, spécifique au système d'exploitation des instances de conteneur. Pour utiliser la méthode sans domaine, une section facultative du fichier CredSpec peut spécifier les informations d'identification dans l'une des options de stockage domainless user credentials du tableau suivant, spécifiques au système d'exploitation des instances de conteneur.

Emplacement de stockage	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	informations d'identification utilisateur sans domaine	informations d'identification utilisateur sans domaine
Boutique de paramètres Amazon EC2 Systems Manager	CredSpec	CredSpec, informations d'identification utilisateur sans domaine
Fichier local	N/A	CredSpec

Prérequis

Avant d'utiliser la fonctionnalité gMSA pour les conteneurs Windows avec Amazon ECS, assurez-vous de remplir les conditions suivantes :

- Vous configurez un domaine Active Directory avec les ressources auxquelles vous souhaitez que vos conteneurs accèdent. Amazon ECS prend en charge les configurations suivantes :
  - Un AWS Directory Service Active Directory. AWS Directory Service est un Active Directory AWS géré hébergé sur Amazon EC2. Pour plus d'informations, consultez Getting Started with AWS Managed Microsoft AD dans le Guide d'AWS Directory Service administration.

  - Un répertoire Active Directory sur site. Vous devez vous assurer que l'instance de conteneur Linux Amazon ECS peut se joindre au domaine. Pour de plus amples informations, veuillez consulter AWS Direct Connect.
- Vous disposez d'un compte gMSA existant dans l'Active Directory. Pour plus d'informations, voir Create g MSAs pour les conteneurs Windows.
- Vous avez choisi d'utiliser gMSA sans domaine ou l'instance de conteneur Amazon ECS Windows hébergeant la tâche Amazon ECS doit être jointe au domaine du répertoire Active Directory et être membre du groupe de sécurité Active Directory qui a accès au compte gMSA.

  En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.
- Vous avez ajouté les autorisations IAM requises. Les autorisations requises dépendent des méthodes que vous choisissez pour les informations d'identification initiales et pour le stockage de la spécification des informations d'identification :
  - Si vous utilisez domainless gMSA pour les informations d'identification initiales, les autorisations IAM pour AWS Secrets Manager sont requises sur le rôle d' EC2instance Amazon.
  - Si vous stockez la spécification des informations d'identification dans SSM Parameter Store, les autorisations IAM pour Amazon EC2 Systems Manager Parameter Store sont requises pour le rôle d'exécution de la tâche.
  - Si vous stockez la spécification des informations d'identification dans Amazon S3, les autorisations IAM d'Amazon Simple Storage Service sont requises sur le rôle d'exécution de la tâche.

Configuration de gMSA pour des conteneurs Windows sur Amazon ECS

Pour configurer gMSA pour les conteneurs Windows sur Amazon ECS, vous pouvez suivre le didacticiel complet qui inclut la configuration des prérequis Utilisation de conteneurs Windows Amazon ECS avec des conteneurs sans domaine à gMSA l'aide du AWS CLI.

Les sections suivantes décrivent la configuration de CredSpec en détail.

Rubriques

- CredSpec exemple
- Configuration de gMSA sans domaine

- Référencement d'un fichier de spécification d'informations d'identification dans une définition de tâche

CredSpec exemple

Amazon ECS utilise un fichier de spécification d'informations d'identification qui contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur Windows. Vous pouvez générer le fichier de spécification d'informations d'identification et le référencer dans le champ `credentialSpec` de votre définition de tâche. Le fichier de spécification d'informations d'identification ne contient pas de secrets.

Voici un exemple de fichier de spécification d'informations d'identification :

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "contoso.com",
        "DnsName": "contoso.com",
        "NetBiosName": "contoso"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "contoso.com"
            }
        ]
    }
}
```

Configuration de gMSA sans domaine

Nous recommandons gMSA sans domaine au lieu de joindre les instances de conteneur à un seul domaine. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour

accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.

1. Avant de charger CredSpec vers l'une des options de stockage, ajoutez des informations au CredSpec à l'aide de l'ARN du secret dans Secrets Manager ou SSM Parameter Store. Pour plus d'informations, consultez la section Configuration des spécifications d'identification supplémentaires pour le cas d'utilisation d' non-domain-joinedun hôte de conteneur sur le site Web de Microsoft Learn.

   Format d'informations d'identification gMSA sans domaine

   Voici le format JSON pour les informations d'identification gMSA sans domaine pour votre Active Directory. Stockez les informations d'identification dans Secrets Manager ou SSM Parameter Store.

   ```
   {
       "username":"WebApp01",
       "password":"Test123!",
       "domainName":"contoso.com"
   }
   ```

2. Ajoutez les informations suivantes au fichier CredSpec contenu dans le `ActiveDirectoryConfig`. Remplacez l'ARN par le secret dans Secrets Manager ou SSM Parameter Store.

   Veuillez noter que la valeur `PluginGUID` doit correspondre au GUID de l'extrait de code suivant et qu'elle est obligatoire.

   ```
       "HostAccountConfig": {
           "PortableCcgVersion": "1",
           "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
           "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
   region:111122223333:secret:gmsa-plugin-input\"}"
       }
   ```

   Vous pouvez également utiliser un secret dans SSM Parameter Store en utilisant l'ARN au format suivant : `\"arn:aws:ssm:aws-region:111122223333:parameter/gmsa-plugin-input\"`.

3. Une fois le fichier CredSpec modifié, il doit ressembler à l'exemple suivant :

```
{
  "CmsPlugins": [
    "ActiveDirectory"
  ],
  "DomainJoinConfig": {
    "Sid": "S-1-5-21-4066351383-705263209-1606769140",
    "MachineAccountName": "WebApp01",
    "Guid": "ac822f13-583e-49f7-aa7b-284f9a8c97b6",
    "DnsTreeName": "contoso",
    "DnsName": "contoso",
    "NetBiosName": "contoso"
  },
  "ActiveDirectoryConfig": {
    "GroupManagedServiceAccounts": [
      {
        "Name": "WebApp01",
        "Scope": "contoso"
      },
      {
        "Name": "WebApp01",
        "Scope": "contoso"
      }
    ],
    "HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
  }
}
```

Référencement d'un fichier de spécification d'informations d'identification dans une définition de tâche

Amazon ECS permet de référencer le chemin du fichier dans le champ `credentialSpecs` d'une définition de tâche. Pour chacune de ces options, vous pouvez fournir `credentialspec:` ou `domainlesscredentialspec:`, selon que vous joignez les instances de conteneur à un seul domaine, ou que vous utilisez gMSA sans domaine, respectivement.

Compartiment Amazon S3

Ajoutez la spécification d'informations d'identification à un compartiment Amazon S3, puis référencez l'Amazon Resource Name (ARN) du compartiment Amazon S3 dans le champ `credentialSpecs` de la définition de tâche.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

Vous devez également ajouter les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche Amazon ECS pour permettre à vos tâches d'accéder au compartiment Amazon S3.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::{bucket_name}",
                "arn:aws:s3:::{bucket_name}/{object}"
            ]
```

```
        }
    ]
}
```

Paramètre SSM Parameter Store

Ajoutez la spécification d'informations d'identification à un compartiment SSM Parameter Store, puis référencez l'Amazon Resource Name (ARN) du paramètre SSM Parameter Store dans le champ `credentialSpecs` de la définition de tâche.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [

 "credentialspecdomainless:arn:aws:ssm:region:111122223333:parameter/parameter_name"
            ],
            ...
        }
    ],
    ...
}
```

Vous devez également ajouter les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche Amazon ECS pour permettre à vos tâches d'accéder au paramètre SSM Parameter Store.

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
```

```
            "Resource": [
            "arn:aws:ssm:arn:aws::111122223333:parameter/parameter_name"
            ]
        }
    ]
}
```

Fichier local

Référencez le chemin d'un fichier local qui contient les détails de la spécification d'informations d'identification dans le champ `credentialSpecs` de la définition de tâche. Le chemin de fichier référencé doit être relatif au répertoire `C:\ProgramData\Docker\CredentialSpecs` et utiliser la barre oblique inverse (« \ ») comme séparateur de chemin de fichier.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspec:file://CredentialSpecDir\CredentialSpecFile.json"
            ],
            ...
        }
    ],
    ...
}
```

Utilisation d' EC2 Image Builder pour créer une solution personnalisée optimisée pour Amazon ECS AMIs

AWS vous recommande d'utiliser les solutions optimisées pour Amazon ECS, AMIs car elles sont préconfigurées avec les exigences et les recommandations nécessaires pour exécuter vos charges de travail de conteneurs. Il peut arriver que vous deviez personnaliser votre AMI pour ajouter des logiciels supplémentaires. Vous pouvez utiliser EC2 Image Builder pour créer, gérer et déployer les images de votre serveur. Vous restez propriétaire des images personnalisées créées dans votre compte. Vous pouvez utiliser les pipelines EC2 Image Builder pour automatiser les mises à jour et les

correctifs système pour vos images, ou utiliser une commande autonome pour créer une image avec les ressources de configuration que vous avez définies.

Vous créez une recette pour votre image. La recette inclut une image parent et tous les composants supplémentaires. Vous créez également un pipeline qui distribue votre AMI personnalisée.

Vous créez une recette pour votre image. Une recette d'image Image Builder est un document qui définit l'image de base et les composants qui sont appliqués à l'image de base afin de produire la configuration souhaitée pour l'image AMI de sortie. Vous créez également un pipeline qui distribue votre AMI personnalisée. Pour plus d'informations, consultez la section Fonctionnement EC2 d'Image Builder dans le guide de l'utilisateur d'EC2 Image Builder.

Nous vous recommandons d'utiliser l'une des images optimisées pour Amazon ECS suivantes AMIs comme « image parent » dans EC2 Image Builder :

- Linux
  - 023 x86 optimisé pour Amazon ECS AL2
  - AMI Amazon Linux 2023 (arm64) optimisée pour Amazon ECS
  - AMI Amazon Linux 2 (noyau 5) optimisée pour Amazon ECS
  - AMI Amazon Linux 2 x86 optimisée pour Amazon ECS
- Windows
  - Windows 2022 Full x86 optimisé pour Amazon ECS
  - Windows 2022 Core x86 optimisé pour Amazon ECS
  - Windows 2019 Full x86 optimisé pour Amazon ECS
  - Windows 2019 Core x86 optimisé pour Amazon ECS
  - Windows 2016 complet x86 optimisé pour Amazon ECS

Nous vous recommandons également de sélectionner « Utiliser la dernière version disponible du système d'exploitation ». Le pipeline utilisera le versionnement sémantique pour l'image parent, ce qui permet de détecter les mises à jour des dépendances dans les tâches planifiées automatiquement. Pour plus d'informations, consultez la section Gestion des versions sémantique dans le guide de l'utilisateur EC2 d'Image Builder.

AWS met régulièrement à jour les images d'AMI optimisées pour Amazon ECS avec des correctifs de sécurité et la nouvelle version de l'agent de conteneur. Lorsque vous utilisez un ID AMI comme image parent dans votre recette d'image, vous devez vérifier régulièrement si des mises à jour ont

été apportées à l'image parent. S'il y a des mises à jour, vous devez créer une nouvelle version de votre recette avec l'AMI mise à jour. Cela garantit que vos images personnalisées incorporent la dernière version de l'image parent. Pour plus d'informations sur la création d'un flux de travail permettant de mettre automatiquement à jour vos EC2 instances dans votre cluster Amazon ECS avec les instances nouvellement créées AMIs, consultez Comment créer un pipeline de renforcement des AMI et automatiser les mises à jour de votre parc d'instances ECS.

Vous pouvez également spécifier le Amazon Resource Name (ARN) d'une image parent publiée via un pipeline EC2 Image Builder géré. Amazon publie régulièrement des images AMI optimisées pour Amazon ECS via des pipelines gérés. Ces images sont accessibles au public. Vous devez disposer des autorisations appropriées pour accéder à l'image. Lorsque vous utilisez un ARN d'image au lieu d'une AMI dans votre recette Image Builder, votre pipeline utilise automatiquement la version la plus récente de l'image parent à chaque exécution. Cette approche élimine le besoin de créer manuellement de nouvelles versions de recettes pour chaque mise à jour.

Utilisation de l'ARN de l'image avec l'infrastructure en tant que code (IaC)

Vous pouvez configurer la recette à l'aide de la console EC2 Image Builder, de l'infrastructure sous forme de code (par exemple AWS CloudFormation) ou du AWS SDK. Lorsque vous spécifiez une image parent dans votre recette, vous pouvez spécifier un ID EC2 AMI, un ARN d'image Image Builder, un identifiant de AWS Marketplace produit ou une image de conteneur. AWS publie publiquement l'image AMI IDs et Image Builder ARNs d'Amazon ECS optimisée AMIs . Le format ARN de l'image est le suivant :

```
arn:${Partition}:imagebuilder:${Region}:${Account}:image/${ImageName}/${ImageVersion}
```

Le format `ImageVersion` est le suivant. Remplacez*`major`*, *`minor`* et *`patch`* par les dernières valeurs.

```
<major>.<minor>.<patch>
```

Vous pouvez remplacer `major` `minor` et `patch` remplacer des valeurs spécifiques ou utiliser l'ARN sans version d'une image, afin que votre pipeline conserve up-to-date la dernière version de l'image parent. Un ARN sans version utilise le format générique « x.x.x » pour représenter la version de l'image. Cette approche permet au service Image Builder de résoudre automatiquement la version la plus récente de l'image. L'utilisation d'un ARN sans version garantit que votre référence pointe toujours vers la dernière image disponible, ce qui rationalise le processus de mise à jour des images dans votre déploiement. Lorsque vous créez une recette avec la console, EC2 Image Builder identifie

automatiquement l'ARN de votre image parent. Lorsque vous utilisez iAC pour créer la recette, vous devez identifier l'ARN et sélectionner la version d'image souhaitée ou utiliser un ARN sans version pour indiquer la dernière image disponible. Nous vous recommandons de créer un script automatique pour filtrer et n'afficher que les images correspondant à vos critères. Le script Python suivant montre comment récupérer une liste d'Amazon ECS AMIs optimisés.

Le script accepte deux arguments facultatifs : `owner` et`platform`, avec les valeurs par défaut « Amazon » et « Windows » respectivement. Les valeurs valides pour l'argument du propriétaire sont : `SelfShared`,`Amazon`, et`ThirdParty`. Les valeurs valides pour l'argument de plate-forme sont `Windows` et`Linux`. Par exemple, si vous exécutez le script avec les `owner` arguments set to `Amazon` et `platform` set to`Linux`, le script génère une liste d'images publiées par Amazon, y compris des images optimisées pour Amazon ECS.

```
import boto3
import argparse

def list_images(owner, platform):
    # Create a Boto3 session
    session = boto3.Session()

    # Create an EC2 Image Builder client
    client = session.client('imagebuilder')

    # Define the initial request parameters
    request_params = {
        'owner': owner,
        'filters': [
            {
                'name': 'platform',
                'values': [platform]
            }
        ]
    }

    # Initialize the results list
    all_images = []

    # Get the initial response with the first page of results
    response = client.list_images(**request_params)

    # Extract images from the response
    all_images.extend(response['imageVersionList'])
```

```
    # While 'nextToken' is present, continue paginating
    while 'nextToken' in response and response['nextToken']:
        # Update the token for the next request
        request_params['nextToken'] = response['nextToken']

        # Get the next page of results
        response = client.list_images(**request_params)

        # Extract images from the response
        all_images.extend(response['imageVersionList'])

    return all_images

def main():
    # Initialize the parser
    parser = argparse.ArgumentParser(description="List AWS images based on owner and
 platform")

    # Add the parameters/arguments
    parser.add_argument("--owner", default="Amazon", help="The owner of the images.
 Default is 'Amazon'.")
    parser.add_argument("--platform", default="Windows", help="The platform type of the
 images. Default is 'Windows'.")

    # Parse the arguments
    args = parser.parse_args()

    # Retrieve all images based on the provided owner and platform
    images = list_images(args.owner, args.platform)

    # Print the details of the images
    for image in images:
        print(f"Name: {image['name']}, Version: {image['version']}, ARN:
 {image['arn']}")

if __name__ == "__main__":
    main()
```

Utilisation de l'ARN de l'image avec AWS CloudFormation

Une recette d'image Image Builder est un plan qui spécifie l'image parent et les composants nécessaires pour obtenir la configuration prévue de l'image de sortie. Vous utilisez la

`AWS::ImageBuilder::ImageRecipe` ressource. Définissez la `ParentImage` valeur sur l'ARN de l'image. Utilisez l'ARN sans version de l'image souhaitée pour vous assurer que votre pipeline utilise toujours la version la plus récente de l'image. Par exemple, `arn:aws:imagebuilder:us-east-1:aws:image/amazon-linux-2023-ecs-optimized-x86/x.x.x`. La définition de `AWS::ImageBuilder::ImageRecipe` ressource suivante utilise un ARN d'image géré par Amazon :

```
ECSRecipe:
    Type: AWS::ImageBuilder::ImageRecipe
    Properties:
      Name: MyRecipe
      Version: '1.0.0'
      Components:
        - ComponentArn: [<The component arns of the image recipe>]
      ParentImage: "arn:aws:imagebuilder:us-east-1:aws:image/amazon-linux-2023-ecs-
optimized-x86/x.x.x"
```

Pour plus d'informations sur cette `AWS::ImageBuilder::ImageRecipe`ressource, consultez le guide de AWS CloudFormation l'utilisateur.

Vous pouvez automatiser la création de nouvelles images dans votre pipeline en définissant la `Schedule` propriété de la `AWS::ImageBuilder::ImagePipeline` ressource. Le calendrier inclut une condition de départ et une expression cron. Pour plus d'informations, consultez `AWS::ImageBuilder::ImagePipeline` dans le Guide de l'utilisateur AWS CloudFormation .

Dans l'`AWS::ImageBuilder::ImagePipeline`exemple suivant, le pipeline exécute une construction à 10 h 00, heure universelle coordonnée (UTC) tous les jours. Définissez les `Schedule` valeurs suivantes :

- Définissez `PipelineExecutionStartCondition` sur `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE`. La génération ne démarre que si des ressources dépendantes telles que l'image parent ou les composants, qui utilisent le caractère générique « x » dans leurs versions sémantiques, sont mises à jour. Cela garantit que la version intègre les dernières mises à jour de ces ressources.
- ScheduleExpression Défini sur l'expression `(0 10 * * ? *)` cron.

```
ECSPipeline:
    Type: AWS::ImageBuilder::ImagePipeline
```

```
    Properties:
      Name: my-pipeline
      ImageRecipeArn: <arn of the recipe you created in previous step>
      InfrastructureConfigurationArn: <ARN of the infrastructure configuration
associated with this image pipeline>
      Schedule:
        PipelineExecutionStartCondition:
EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE
        ScheduleExpression: 'cron(0 10 * * ? *)'
```

Utilisation de l'ARN de l'image avec Terraform

L'approche permettant de spécifier l'image parent et le calendrier de votre pipeline dans Terraform s'aligne sur celle de. AWS CloudFormation Vous utilisez la `aws_imagebuilder_image_recipe` ressource. Définissez la `parent_image` valeur sur l'ARN de l'image. Utilisez l'ARN sans version de l'image souhaitée pour vous assurer que votre pipeline utilise toujours la version la plus récente de l'image. Pour plus d'informations, consultez `aws_imagebuilder_image_recipe`la documentation Terraform.

Dans le bloc de configuration de planification du`aws_imagebuilder_image_pipeline resource`, définissez la valeur de l'`schedule_expression`argument sur une expression cron de votre choix pour spécifier la fréquence d'exécution du pipeline, puis définissez la `pipeline_execution_start_condition` valeur sur. `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE` Pour plus d'informations, consultez `aws_imagebuilder_image_pipeline`la documentation Terraform.

Utilisation de AWS Deep Learning Containers sur Amazon ECS

AWS Les Deep Learning Containers fournissent un ensemble d'images Docker pour la formation et le service de modèles dans TensorFlow et Apache MXNet (incubation) sur Amazon ECS. Les Deep Learning Containers permettent d'optimiser les environnements grâce TensorFlow aux bibliothèques NVIDIA CUDA (pour les instances GPU) et Intel MKL (pour les instances CPU). Les images de conteneur pour les conteneurs Deep Learning Containers sont disponibles dans Amazon ECR pour être référencés dans les définitions de tâches Amazon ECS. Vous pouvez utiliser les conteneurs Deep Learning Containers avec Amazon Elastic Inference pour réduire vos coûts d'inférence.

Pour commencer à utiliser les Deep Learning Containers sans Elastic Inference sur Amazon ECS, consultez la section relative à la configuration d'Amazon ECS dans le guide du AWS Apprentissage profond (deep learning) AMIs développeur.

Quotas de service Amazon ECS service

Lorsque vous créez votre Compte AWS, nous définissons des quotas par défaut (également appelés limites) pour vos AWS ressources par région. Si vous tentez de dépasser le quota d'une ressource, la demande échoue.

La console Service Quotas est un emplacement central où vous pouvez consulter et gérer vos quotas de AWS services, et demander une augmentation des quotas pour la plupart des ressources que vous utilisez. Utilisez les informations relatives que nous fournissons pour gérer vos ressources. Prévoyez de demander les augmentations de quota avant le moment où vous en aurez besoin.

Pour plus d'informations sur les quotas Amazon ECS et Fargate, consultez la section Points de terminaison et quotas Amazon ECS dans le. Référence générale d'Amazon Web Services Pour plus d'informations sur la limitation des API dans l'API Amazon ECS, consultez Limitation des demandes pour l'API Amazon ECS.

Gestion de votre Amazon ECS et AWS Fargate de vos quotas de service dans AWS Management Console

Amazon ECS a intégré Service Quotas, un AWS service qui vous permet de consulter et de gérer vos quotas depuis un emplacement central. Pour plus d'informations, veuillez consulter la rubrique Qu'est-ce que Service Quotas ? dans le Guide de l'utilisateur Service Quotas.

Service Quotas facilite la recherche de la valeur de vos quotas de service Amazon ECS service.

AWS Management Console

Pour afficher les quotas de service Amazon ECS et Fargate à l'aide de la AWS Management Console

1. Ouvrez la console Service Quotas sur https://console.aws.amazon.com/servicequotas/.
2. Dans le panneau de navigation, choisissez Services AWS .
3. Dans la liste services (Services AWS), recherchez et sélectionnez Amazon Elastic Container Service (Amazon ECS) ou AWS Fargate.

   Dans la liste Service quotas (Quotas de service), vous pouvez voir le nom du Service Quota, la valeur appliquée (le cas échéant), le quota AWS par défaut et si la valeur du quota est réglable.

4. Pour afficher des informations supplémentaires sur un quota de service, notamment la description, choisissez le nom du quota.
5. (Facultatif) Pour demander une augmentation de quota, sélectionnez le quota que vous souhaitez augmenter, sélectionnez Request quota increase (Demander une augmentation de quota), saisissez ou sélectionnez les informations requises, puis sélectionnez Request (Demander).

Pour travailler davantage avec les quotas de service à l'aide du AWS Management Console guide de l'utilisateur sur les quotas de service. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

AWS CLI

Pour afficher les quotas de service Amazon ECS et Fargate à l'aide de la AWS CLI

Exécutez la commande suivante pour afficher les quotas Amazon ECS par défaut.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code ecs \
    --output table
```

Exécutez la commande suivante pour afficher les quotas Fargate par défaut.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code fargate \
    --output table
```

Exécutez la commande suivante pour afficher les quotas Fargate appliqués.

```
aws service-quotas list-service-quotas \
    --service-code fargate
```

> **Note**
>
> Amazon ECS ne prend pas en charge les quotas appliqués.

Pour plus d'informations sur l'utilisation des quotas de service à l'aide du AWS CLI, consultez le Guide de référence des AWS CLI commandes Service Quotas. Pour demander une augmentation de quota, consultez la commande `request-service-quota-increase` dans la référence des commandes AWS CLI.

Gérez les quotas de service Amazon ECS et les limites de limitation des API

Amazon ECS est intégré à plusieurs d'entre Services AWS eux, notamment Elastic Load Balancing et Amazon EC2. AWS Cloud Map Grâce à cette intégration étroite, Amazon ECS inclut plusieurs fonctionnalités telles que l'équilibrage de charge des services, Service Connect, la mise en réseau des tâches et le dimensionnement automatique des clusters. Amazon ECS et l'autre solution Services AWS qu'il intègre à tous les systèmes maintiennent les quotas de service et les limites de débit des API afin de garantir des performances et une utilisation cohérentes. Ces quotas de service empêchent également le provisionnement accidentel de ressources supérieures à celles nécessaires et protègent contre les actions malveillantes susceptibles d'augmenter votre facture.

En vous familiarisant avec vos quotas de service et les limites de débit des AWS API, vous pouvez planifier le dimensionnement de vos charges de travail sans vous soucier d'une dégradation inattendue des performances. Pour plus d'informations, consultez la section Régulation des demandes pour l'API Amazon ECS.

Lorsque vous dimensionnez vos charges de travail sur Amazon ECS, nous vous recommandons de prendre en compte le quota de service suivant.

- AWS Fargate possède des quotas qui limitent le nombre de tâches exécutées simultanément dans chacune d'elles Région AWS. Il existe des quotas pour les tâches On-Demand et Fargate Spot sur Amazon ECS. Chaque quota de service inclut également tous les pods Amazon EKS que vous utilisez sur Fargate.
- Pour les tâches exécutées sur des EC2 instances Amazon, le nombre maximum d' EC2 instances Amazon que vous pouvez enregistrer pour chaque cluster est de 5 000. Si vous utilisez le dimensionnement automatique du cluster Amazon ECS avec un fournisseur de capacité de groupe Auto Scaling, ou si vous gérez vous-même les EC2 instances Amazon de votre cluster, ce quota peut devenir un obstacle au déploiement. Si vous avez besoin de plus de capacité, vous pouvez créer d'autres clusters ou demander une augmentation du quota de service.

- Si vous utilisez le dimensionnement automatique du cluster Amazon ECS avec un fournisseur de capacité de groupe Auto Scaling, tenez compte du `Tasks in the PROVISIONING state per cluster` quota lors du dimensionnement de vos services. Ce quota est le nombre maximum de tâches dans l'`PROVISIONING`état pour chaque cluster pour lesquelles les fournisseurs de capacité peuvent augmenter la capacité. Lorsque vous lancez un grand nombre de tâches simultanément, vous pouvez facilement atteindre ce quota. Par exemple, si vous déployez simultanément des dizaines de services, chacun comportant des centaines de tâches. Dans ce cas, le fournisseur de capacité doit lancer de nouvelles instances de conteneur pour placer les tâches lorsque la capacité du cluster est insuffisante. Pendant que le fournisseur de capacité lance des EC2 instances Amazon supplémentaires, le planificateur de services Amazon ECS continuera probablement à lancer des tâches en parallèle. Toutefois, cette activité peut être limitée en raison d'une capacité de cluster insuffisante. Le planificateur de services Amazon ECS met en œuvre une stratégie de ralentissement et de limitation exponentielle pour réessayer de placer des tâches lorsque de nouvelles instances de conteneur sont lancées. Par conséquent, les délais de déploiement ou de mise à l'échelle peuvent être plus lents. Pour éviter cette situation, vous pouvez planifier vos déploiements de services selon l'une des méthodes suivantes. Déployez un grand nombre de tâches sans augmenter la capacité du cluster ou conservez une capacité de cluster inutilisée pour le lancement de nouvelles tâches.

En plus de prendre en compte le quota de service Amazon ECS lors du dimensionnement de vos charges de travail, tenez également compte du quota de service pour Services AWS les autres services intégrés à Amazon ECS.

Elastic Load Balancing

Vous pouvez configurer vos services Amazon ECS pour utiliser Elastic Load Balancing afin de répartir le trafic de manière uniforme entre les tâches. Pour plus d'informations et les meilleures pratiques recommandées concernant le choix d'un équilibreur de charge, consultez[Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS](#).

Quotas du service Elastic Load Balancing

Lorsque vous dimensionnez vos charges de travail, tenez compte des quotas du service Elastic Load Balancing suivants. La plupart des quotas du service Elastic Load Balancing sont ajustables, et vous pouvez demander une augmentation dans la console Service Quotas.

Application Load Balancer

Lorsque vous utilisez un Application Load Balancer, selon votre cas d'utilisation, vous devrez peut-être demander une augmentation de quota pour :

- Le `Targets per Application Load Balancer` quota, c'est-à-dire le nombre de cibles situées derrière votre Application Load Balancer.
- Le `Targets per Target Group per Region` quota, c'est-à-dire le nombre de cibles derrière vos groupes cibles.

Pour plus d'informations, consultez la section Quotas pour vos équilibreurs de charge d'application dans le Guide de l'utilisateur pour les équilibreurs de charge d'application.

Network Load Balancer

Le nombre de cibles que vous pouvez enregistrer auprès d'un Network Load Balancer est plus limité. Lorsque vous utilisez un Network Load Balancer, vous souhaiterez souvent activer le support entre zones, ce qui implique des limites de dimensionnement supplémentaires quant `Targets per Availability Zone Per Network Load Balancer` au nombre maximum de cibles par zone de disponibilité pour chaque Network Load Balancer. Pour plus d'informations, consultez la section Quotas pour vos équilibreurs de charge réseau dans le Guide de l'utilisateur des équilibreurs de charge réseau.

Limitation de l'API Elastic Load Balancing

Lorsque vous configurez un service Amazon ECS pour utiliser un équilibreur de charge, les contrôles de santé du groupe cible doivent réussir avant que le service ne soit considéré comme sain. Pour effectuer ces contrôles de santé, Amazon ECS invoque les opérations de l'API Elastic Load Balancing en votre nom. Si vous avez configuré un grand nombre de services avec des équilibreurs de charge dans votre compte, vous risquez de ralentir les déploiements de services en raison d'un éventuel ralentissement spécifique aux opérations de l'API `RegisterTargetDeregisterTarget`, et d'`DescribeTargetHealth`Elastic Load Balancing. En cas de régulation, des erreurs de régulation apparaissent dans les messages d'événements de votre service Amazon ECS.

Si vous êtes confronté à une limitation d' AWS Cloud Map API, vous pouvez nous contacter Support pour obtenir des conseils sur la manière d'augmenter vos limites de limitation AWS Cloud Map d'API. Pour plus d'informations sur la surveillance et le dépannage de telles erreurs de régulation, consultez. Gérer les problèmes de régulation d'Amazon ECS

Interfaces réseau Elastic

Lorsque vos tâches utilisent le mode `awsvpc` réseau, Amazon ECS fournit une interface Elastic Network (ENI) unique pour chaque tâche. Lorsque vos services Amazon ECS utilisent un équilibreur de charge Elastic Load Balancing, ces interfaces réseau sont également enregistrées en tant que cibles pour le groupe cible approprié défini dans le service.

Quotas de service d'interface réseau élastiques

Lorsque vous exécutez des tâches utilisant le mode `awsvpc` réseau, une interface Elastic Network unique est associée à chaque tâche. Si ces tâches doivent être accessibles via Internet, attribuez une adresse IP publique à l'interface elastic network pour ces tâches. Lorsque vous dimensionnez vos charges de travail Amazon ECS, tenez compte de ces deux quotas importants :

- Le `Network interfaces per Region` quota qui est le nombre maximum d'interfaces réseau dans et Région AWS pour votre compte.
- Le `Elastic IP addresses per Region` quota qui est le nombre maximum d'adresses IP élastiques dans un Région AWS.

Ces deux quotas de service sont ajustables et vous pouvez demander une augmentation depuis votre console Service Quotas pour ces derniers. Pour plus d'informations, consultez les quotas de service Amazon VPC dans le guide de l'utilisateur d'Amazon Virtual Private Cloud.

Pour les charges de travail Amazon ECS hébergées sur des EC2 instances Amazon, lors de l'exécution de tâches utilisant le mode `awsvpc` réseau, tenez compte du quota de `Maximum network interfaces` service, c'est-à-dire du nombre maximum d'instances réseau pour chaque EC2 instance Amazon. Ce quota limite le nombre de tâches que vous pouvez placer sur une instance. Vous ne pouvez pas ajuster le quota et celui-ci n'est pas disponible dans la console Service Quotas. Pour plus d'informations, consultez la section Adresses IP par interface réseau et par type d'instance dans le guide de EC2 l'utilisateur Amazon.

Bien que vous ne puissiez pas modifier le nombre d'interfaces réseau pouvant être associées à une EC2 instance Amazon, vous pouvez utiliser la fonctionnalité Elastic Network Interface trunking pour augmenter le nombre d'interfaces réseau disponibles. Par exemple, par défaut, une `c5.large` instance peut avoir jusqu'à trois interfaces réseau. L'interface réseau principale de l'instance est considérée comme une interface réseau. Vous pouvez donc associer deux interfaces réseau supplémentaires à l'instance. Étant donné que chaque tâche utilisant le mode `awsvpc` réseau nécessite une interface réseau, vous ne pouvez généralement exécuter que deux tâches de ce type

sur ce type d'instance. Cela peut entraîner une sous-utilisation de la capacité de votre cluster. Si vous activez Elastic Network Interface trunking, vous pouvez augmenter la densité de l'interface réseau afin de placer un plus grand nombre de tâches sur chaque instance. Lorsque le trunking est activé, une `c5.large` instance peut avoir jusqu'à 12 interfaces réseau. L'instance possède l'interface réseau principale et Amazon ECS crée et attache une interface réseau « tronc » à l'instance. Par conséquent, avec cette configuration, vous pouvez exécuter 10 tâches sur l'instance au lieu des deux tâches par défaut. Pour de plus amples informations, veuillez consulter Augmenter les interfaces réseau des instances de conteneur Linux Amazon ECS.

Limitation des API d'interface réseau élastique

Lorsque vous exécutez des tâches qui utilisent le mode `awsvpc` réseau, Amazon ECS s'appuie sur les Amazon suivants EC2 APIs. Chacun d'entre eux APIs a des limites d'API différentes. Pour plus d'informations, consultez la section Régulation des demandes pour l' EC2 API Amazon dans le manuel Amazon EC2 API Reference.

- CreateNetworkInterface
- AttachNetworkInterface
- DetachNetworkInterface
- DeleteNetworkInterface
- DescribeNetworkInterfaces
- DescribeVpcs
- DescribeSubnets
- DescribeSecurityGroups
- DescribeInstances

Si les appels d' EC2 API Amazon sont limités pendant les flux de travail de provisionnement de l'interface Elastic Network Interface, le planificateur de services Amazon ECS réessaie automatiquement avec des interruptions exponentielles. Ces mises hors service automatiques peuvent parfois retarder le lancement des tâches, ce qui ralentit les vitesses de déploiement. En cas de limitation de l'API, le message apparaît `Operations are being throttled. Will try again later.` sur vos messages d'événement de service. Si vous respectez régulièrement les limites des EC2 API Amazon, vous pouvez nous contacter Support pour obtenir des conseils sur la manière d'augmenter vos limites de limitation des API. Pour plus d'informations sur la surveillance et le dépannage des erreurs de régulation, consultez la section Gestion des problèmes de régulation.

AWS Cloud Map

Amazon ECS Service Discovery et Service Connect sont utilisés AWS Cloud Map APIs pour gérer les espaces de noms de vos services Amazon ECS. Si vos services comportent un grand nombre de tâches, tenez compte des recommandations suivantes.

AWS Cloud Map quotas de service

Lorsque les services Amazon ECS sont configurés pour utiliser Service Discovery ou Service Connect, le `Tasks per service` quota, qui est le nombre maximum de tâches pour le service, est affecté par le quota de AWS Cloud Map `Instances per service` service, qui est le nombre maximum d'instances pour ce service. En particulier, le quota de AWS Cloud Map service réduit le nombre d'instances que vous pouvez exécuter à un maximum de 1 000 instances par service. Vous ne pouvez pas modifier le AWS Cloud Map quota. Pour plus d'informations, consultez Quotas de service AWS Cloud Map.

AWS Cloud Map Limitation de l'API

Amazon ECS appelle le`ListInstances`, `GetInstancesHealthStatusRegisterInstance`, et `DeregisterInstance` AWS Cloud Map APIs en votre nom. Ils facilitent la découverte des services et effectuent des bilans de santé lorsque vous lancez une tâche. Lorsque plusieurs services utilisant la découverte de services comportant un grand nombre de tâches sont déployés en même temps, cela peut entraîner le dépassement des limites de limitation des AWS Cloud Map API. Dans ce cas, vous verrez probablement le message suivant : `Operations are being throttled. Will try again later` dans votre service Amazon ECS, des messages d'événement indiquant un ralentissement du déploiement et de la vitesse de lancement des tâches. AWS Cloud Map ne documente pas les limites de limitation pour ceux-ci. APIs Si vous êtes confronté à un ralentissement dû à ces derniers, vous pouvez nous contacter Support pour obtenir des conseils sur l'augmentation des limites de limitation de vos API. Pour plus de recommandations sur la surveillance et le dépannage de telles erreurs de régulation, consultez. Gérer les problèmes de régulation d'Amazon ECS

Référence d'API Amazon ECS

Outre le AWS Management Console et le AWS Command Line Interface (AWS CLI), Amazon ECS fournit également une API. Vous pouvez utiliser l'API pour automatiser les tâches de gestion des ressources Amazon ECS.

- Pour obtenir la liste des opérations d'API par ressource Amazon ECS, veuillez consulter Actions par ressource Amazon ECS (langue française non garantie).
- Pour obtenir la liste alphabétique des opérations d'API, consultez Actions.
- Pour obtenir la liste alphabétique des types de données, consultez Types de données.
- Pour consulter la liste des paramètres de requête courants, reportez-vous à la page Paramètres courants.
- Pour la description des codes d'erreur, veuillez consulter la page Erreurs courantes.

Pour plus d'informations à ce sujet AWS CLI, consultez la AWS Command Line Interface référence relative à Amazon ECS.

Historique du document

Le tableau suivant décrit les principales mises à jour et les nouvelles fonctions pour le Guide du développeur Amazon Elastic Container Service. Nous mettons aussi la documentation à jour régulièrement pour prendre en compte les commentaires qui nous sont envoyés.

Modification	Description	Date
Ajoutez des autorisations à la politique gérée par Amazon EC2 Container Servicefor EC2 Role.	La politique IAM `AmazonEC2ContainerServiceforEC2Role` gérée a été mise à jour pour inclure l'`ecs:ListTagsForResource` autorisation. Cette autorisation permet à l'agent Amazon ECS de récupérer les balises d'instance de tâche et de conteneur via le point de terminaison des métadonnées des tâches (`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`). Pour plus d'informations, consultez Amazon EC2 ContainerServicefor EC2 Role.	4 août 2025
Ajoutez des autorisations à Amazon ECSInfrastructure RolePolicyForLoadBalancers.	La politique IAM `AmazonECSInfrastructureRolePolicyForLoadBalancers` gérée a été mise à jour avec de nouvelles autorisations pour la description, le désenregistrement et l'enregistrement des groupes cibles.	25 juillet 2025
Support pour la mise à jour d'un contrôleur de déploiement de services	Amazon ECS prend en charge la mise à jour d'un contrôleur de déploiement de services, ce qui vous permet de passer d'un type de déploiement à un autre pour un service existant. Pour plus d'informations, consultez Mettre à jour les paramètres du service Amazon ECS.	15 juillet 2025
blue/green Déploiements natifs d'Amazon ECS	Amazon ECS prend en charge les blue/green déploiements natifs, ce qui vous permet de valider les nouvelles révisions de service avant de diriger le trafic de production vers celles-ci. Pour plus d'informations,	15 juillet 2025

Modification	Description	Date
	consultez la section blue/green Déploiements Amazon ECS.	
Ajoutez une nouvelle politique ECSInfrastructure RolePolicyForLoadBalancers gérée par Amazon.	Cette politique donne accès aux autres ressources de AWS service nécessaires pour gérer les équilibreurs de charge associés aux charges de travail Amazon ECS en votre nom. Pour plus d'informations, consultez Amazon ECSInfrastructure RolePolicyForLoadBalancers.	15 juillet 2025
Support du manifeste d'index SOCI v2	Les clients de Fargate souhaitant utiliser SOCI ne pourront utiliser que le manifeste d'index SOCI v2. Les clients existants qui ont déjà utilisé SOCI sur Fargate peuvent continuer à utiliser le manifeste d'index SOCI v1, mais nous leur conseillons vivement de migrer vers le manifeste d'index SOCI v2. Pour plus d'informations, consultez la section Chargement différé d'images de conteneurs à l'aide de Seekable OCI (SOCI).	2 juillet 2025
Ajout de l'ID de tâche dans les événements d'action de service générés en raison de défaillances de santé.	Pour plus d'informations, consultez les messages relatifs aux événements relatifs à un dysfonctionnement du service Amazon ECS.	30 juin 2025
Passez au mode pilote de journal par défaut.	Le mode de journalisation par défaut pour Amazon ECS a été mis à jour de `blocking` à `non-blocking`. Vous pouvez remplacer ce mode par défaut en spécifiant un `mode` dans votre définition de conteneur `logConfiguration` ou configurer votre propre mode par défaut en définissant les paramètres du `defaultLogDriverMode` compte à l'aide de la console Amazon ECS ou du AWS CLI. Pour plus d'informations, consultez la section Mode pilote de journal par défaut.	25 juin 2025

Modification	Description	Date
Ajoutez des autorisations à Amazon ECSService RolePolicy.	La politique IAM `AmazonECSServiceRolePolicy` gérée a été mise à jour avec de nouvelles AWS Cloud Map autorisations permettant à Amazon ECS de mettre à jour les attributs de AWS Cloud Map service pour les services gérés par Amazon ECS.	24 juin 2025
Support de l'AMI complète Windows Server 2025 optimisée pour Amazon ECS et de l'AMI Windows Server 2025 Core optimisée pour Amazon ECS.	Pour plus d'informations, consultez Windows optimisé pour Amazon ECS. AMIs	20 juin 2025
Support à la mise à jour d'une stratégie de fournisseur de capacité de service.	Vous pouvez désormais mettre à jour une stratégie de fournisseur de capacité de service. Pour plus d'informations, consultez la section Mettre à jour les paramètres du service.	12 juin 2025
Ajouter des autorisations à Amazon ECSInfrastructure RolePolicyForVolumes	La `AmazonECSInfrastructureRolePolicyForVolumes` politique a été mise à jour pour ajouter l'`ec2:DescribeInstances` autorisation. Cette autorisation permet d'éviter les collisions de noms d'appareils pour les volumes Amazon EBS attachés à des tâches Amazon ECS exécutées sur la même instance de conteneur.	2 juin 2025
Support du taux d'initialisation des volumes Amazon EBS associés aux tâches Amazon ECS.	Vous pouvez désormais définir un `volumeInitializationRate` pour déterminer le débit, en Mbits/s, auquel les données sont extraites d'un instantané d'un volume Amazon EBS existant afin de créer de nouveaux volumes à associer à des tâches Amazon ECS. Pour de plus amples informations, veuillez consulter volumeInitializationRate.	13 mai 2025

Modification	Description	Date
Support pour arrêter le déploiement d'un service.	Vous pouvez désormais arrêter le déploiement d'un service. Pour plus d'informations, consultez Arrêter les déploiements de services Amazon ECS.	2 mai 2025
Support pour les points de terminaison à double pile Amazon ECS.	Vous pouvez désormais utiliser des points de terminais on à double pile pour interagir avec Amazon ECS. Les points de terminaison à double pile prennent en charge les demandes adressées à Amazon ECS via le protocole Internet version 4 (IPv4) et le protocole Internet version 6 (IPv6). Pour plus d'informations, consultez la section Utilisation des points de terminais on à double pile Amazon ECS.	30 avril 2025
Support pour configurer un mode pilote de journal par défaut au niveau du compte dans un Région AWS.	Vous pouvez désormais configurer un mode de livraison par défaut des messages de journal depuis un conteneur vers le pilote de journal que vous spécifiez dans `logConfiguration` les paramètres de votre `defaultLogDriverMode` compte. Pour plus d'informations, consultez la section Mode pilote de journal par défaut.	17 avril 2025
Support pour la migration d'un service au format ARN court vers le format ARN long.	Vous pouvez désormais migrer un service au format ARN court vers le paquet au format ARN long. Pour plus d'informations, consultez Migrer un ARN de service court Amazon ECS vers un ARN long.	13 février 2025
Support pour trois types d'injection de pannes sur Amazon ECS et Fargate.	Vous pouvez désormais utiliser le port blackhole du réseau, la latence réseau et l'injection d'erreurs de perte de paquets réseau avec Amazon ECS et Fargate. Pour plus d'informations, consultez Utiliser AWS Fault Injection Service pour tester vos charges de travail Amazon ECS.	19 décembre 2024

Modification	Description	Date
Support pour CloudWatch Container Insights avec une meilleure observabilité.	Vous pouvez utiliser CloudWatch Container Insights avec une meilleure observabilité. Pour plus d'informations, consultez Surveiller les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée.	2 décembre 2024
Support pour la mise à l'échelle prédictive.	Vous pouvez utiliser la mise à l'échelle prédictive pour anticiper les besoins futurs et augmenter de manière proactive les tâches de votre service en fonction des besoins. Pour plus d'informations, voir Utiliser des modèles historiques pour dimensionner les CloudWatch services grâce à une mise à l'échelle prédictive.	21 novembre 2024
Support pour le rééquilibrage des services.	Amazon ECS peut désormais équilibrer automatiquement vos tâches de service entre les zones de disponibilité. Pour de plus amples informations, veuillez consulter Équilibrer un service Amazon ECS entre les zones de disponibilité.	19 novembre 2024
Support pour configurer la résolution des balises d'image du conteneur par rapport aux résumés d'images.	Vous pouvez désormais configurer la résolution des balises d'image de conteneur en résumés pour chaque conteneur dans les tâches faisant partie d'un service. Pour plus d'informations, consultez Résolution de l'image du conteneur et `versionConsistency`.	19 novembre 2024
Ajouter une nouvelle `AmazonECSInfrastructureRolePolicyForVpcLattice` politique IAM	La `AmazonECSInfrastructureRolePolicyForVpcLattice` politique donne accès aux autres ressources de AWS service nécessaires pour gérer en votre nom la fonctionnalité VPC Lattice dans les charges de travail Amazon ECS.	18 novembre 2024

Modification	Description	Date
VPC Lattice est désormais disponible pour les services Amazon ECS.	Vous pouvez désormais utiliser VPC Lattice pour standardiser votre service-to-service connectiv ité, votre sécurité et votre observabilité. Pour plus d'informations, consultez Utiliser Amazon VPC Lattice pour connecter, observer et sécuriser vos services Amazon ECS.	18 novembre 2024
Support de l'historique des services à l'aide de déploiements de services et de révisions de services.	Vous pouvez utiliser les déploiements de services et les révisions de services pour consulter l'histori que des déploiements. Pour plus d'informations, consultez Afficher l'historique des services à l'aide des déploiements de services Amazon ECS et des révisions des services Amazon ECS.	30 octobre 2024
Support pour les volumes Amazon EBS pour le type de EC2 lancement utilisant le système d'exploitation Windows.	Amazon ECS prend en charge les volumes Amazon EBS associés à des tâches exécutées sur EC2 des instances avec le système d'exploitation Windows. Pour plus d'informations, consultez Utiliser les volumes Amazon EBS avec Amazon ECS.	24 octobre 2024
Ajoutez des autorisat ions à Amazon ECSInfras tructureRolePolicy ForVolumes.	La `AmazonECSInfrastructureRolePolicyForVolumes` politique a été mise à jour pour permettre aux clients de créer un volume Amazon EBS à partir d'un instantané. Pour plus d'informa tions, consultez Amazon ECSInfrastructure RolePolic yForVolumes.	10 octobre 2024
Fargate Spot est disponible pour Linux ARM.	Support pour Fargate Spot sur Linux ARM a été ajouté à Amazon ECS.	6 septembre 2024

Modification	Description	Date
Politique de redémarrage des conteneurs pour les conteneurs individuels dans les tâches Amazon ECS.	Vous pouvez activer les politiques de redémarrage pour les conteneurs essentiels et non essentiels définies dans votre définition de tâche afin de surmonter plus rapidement les défaillances transitoires et de maintenir la disponibilité des tâches. Pour de plus amples informations, veuillez consulter Redémarrer des conteneurs individuels dans des tâches Amazon ECS avec des politiques de redémarrage de conteneurs.	15 août 2024
Autorisations ajoutées à the section called “Amazon ECS_ FullAccess”	La `AmazonECS_FullAccess` politique a été mise à jour pour ajouter `iam:PassRole` des autorisations pour les rôles IAM pour un rôle nommé`ecsInfrastructureRole`. Il s'agit du rôle IAM par défaut créé par le AWS Management Console et destiné à être utilisé comme rôle d'infrastructure ECS permettant à Amazon ECS de gérer les volumes Amazon EBS attachés aux tâches ECS.	13 août 2024
Ajout de la prise en charge d'Amazon Linux 2023, de CentOS Stream 9, de Fedora 40 et d'Ubuntu 24 pour ECS Anywhere	Support pour les systèmes d'exploitation Amazon Linux 2023, CentOS Stream 9, Fedora 40 et Ubuntu 24 a été ajouté à ECS Anywhere. Pour de plus amples informations, veuillez consulter Systèmes d'exploitation et architectures système pris en charge.	23 juillet 2024
Résolution d'image de conteneur pour les services utilisant le type de déploiement de mise à jour continue (ECS).	Pour garantir que toutes les tâches d'un service qui utilise le contrôleur de déploiement des mises à jour continues utilisent la même image de conteneur, Amazon ECS convertit les noms des images de conteneur et les balises d'image spécifiées dans la définition de la tâche en résumés d'images de conteneurs. Pour plus d'informations, consultez la section Résolution d'image du conteneur.	10 juillet 2024

Modification	Description	Date
Support de Debian 11 et Debian 12 ajouté pour ECS Anywhere	Support pour les systèmes d'exploitation Debian 11 et Debian 12 a été ajouté à ECS Anywhere. Pour de plus amples informations, veuillez consulter Systèmes d'exploitation et architectures système pris en charge.	28 mars 2024
GMSA pour les conteneur s Linux sur le support de Fargate	Amazon ECS prend en charge l'authentification Active Directory pour les conteneurs Linux sur Fargate via un type spécial de compte de service appelé compte de service géré de groupe (GMSA). Pour plus d'informa tions, consultez la section Utilisation gMSA pour les Linux conteneurs sur Fargate.	5 mars 2024
CloudWatch métriques ajoutées pour les volumes Amazon EBS associés à des tâches	Amazon ECS publie désormais des CloudWatc h métriques relatives à l'utilisation du stockage Amazon EBS pour les tâches associées à un volume Amazon EBS. Pour plus d'informations, consultez les CloudWatch métriques Amazon ECS.	8 février 2024
Service Connect TLS	Vous pouvez désormais utiliser le protocole TLS avec Service Connect.	22 janvier 2024
Politique gérée par Service Connect TLS	Ajout d'une nouvelle ECSInfrastructure RolePolic yForServiceConnectTransportLayerSecurity politique Amazon.	22 janvier 2024
Mise à jour de la configuration du délai d'expiration de Service Connect	La configuration du délai d'expiration de Service Connect peut désormais être mise à jour et inclut deux paramètres facultatifs : `idleTimeout` `etperRequestTimeout` .	22 janvier 2024
Drainage d'instance géré par Amazon ECS	Vous pouvez utiliser le drainage d'instance géré par Amazon ECS pour faciliter la résiliation progressive des instances Amazon ECS.	19 janvier 2024

Modification	Description	Date
Ajout de la prise en charge d'Ubuntu 22 pour ECS Anywhere	Support pour le système d'exploitation Ubuntu 22 a été ajouté à ECS Anywhere. Pour de plus amples informations, veuillez consulter Systèmes d'exploitation et architectures système pris en charge.	16 janvier 2024
Ajouter une `AmazonECSInfrastructureRolePolicyForVolumes` politique IAM	Le `AmazonECSInfrastructureolePolicyForVolumes` a été ajouté. La politique accorde les autorisations nécessaires à Amazon ECS pour effectuer des appels d' AWS API afin de gérer les volumes Amazon EBS associés aux charges de travail Amazon ECS.	11 janvier 2024
Volume de données Amazon EBS pour la tâche Amazon ECS	Vous pouvez configurer 1 volume de données Amazon EBS par tâche pendant le déploiement pour le rattacher à des tâches Amazon ECS autonomes ou à des tâches gérées par un service ECS. La configuration d'un volume lors du déploiement vous permet de créer des définitions de tâches réutilisables qui ne sont pas limitées à des types ou paramètres de volume spécifiques. Les volumes Amazon EBS fournissent un stockage par blocs hautement disponible, rentable, durable et performant pour les charges de travail conteneurisées gourmandes en données.	11 janvier 2024
La console Amazon ECS Classic est arrivée en fin de vie	La console Amazon ECS a atteint la fin de son cycle de vie.	4 décembre 2023
Stratégie mise à jour	La politique IAM ECSService RolePolicy gérée par Amazon a été mise à jour avec de nouvelles `events` autorisations `autoscaling` et `autoscaling-plans` autorisations supplémentaires.	4 décembre 2023

Modification	Description	Date
Support de surveillance du temps d'exécution	Vous pouvez utiliser Runtime Monitoring pour surveiller vos charges de travail Amazon ECS afin d'identifier les comportements malveillants ou non autorisés. Pour plus d'informations, consultez la section Surveillance du temps d'exécution.	26 novembre 2023
Stratégie mise à jour	La politique IAM `AmazonECSServiceRolePolicy` gérée a été mise à jour pour autoriser l'accès à l' AWS Cloud Map `DiscoverInstancesRevision` API.	4 octobre 2023
AWS Fargate configuration de retrait des tâches	Vous pouvez configurer la période d'attente avant le retrait des tâches Fargate. Pour plus d'informations, veuillez consulter Maintenance des tâches AWS Fargate (langue française non garantie).	5 septembre 2023
Paramètres de définition de tâches supplémentaires dans AWS Fargate	AWS Fargate ajoute le support pour `pidMode` et `systemControls` dans la version de la plate-forme Linux`1.4.0`. Pour plus d'informations, veuillez consulter Définitions de tâches (langue française non garantie).	9 août 2023
Refonte de la page de définition des tâches de la console Amazon ECS	La page de définition des tâches de la console Amazon ECS a été repensée et contient désormais des options supplémentaires. Pour plus d'informations, veuillez consulter Création d'une définition de tâche à l'aide de la console (langue française non garantie).	26 juillet 2023

Modification	Description	Date
Fargate prend en charge le chargement différé avec les index Seekable OCI.	AWS Fargate introduit les index SEEKABLE OCI (SOCI). Avec SOCI, les conteneurs ne passent que quelques secondes à extraire l'image avant de pouvoir démarrer, ce qui laisse le temps de configurer l'environnement et d'instancier l'application pendant que l'image est téléchargée en arrière-plan. Pour plus d'informations, consultez la section Chargement différé d'images de conteneurs à l'aide de Seekable OCI (SOCI).	17 juillet 2023
Prise en charge améliorée pour gMSA sous Linux et Windows	La définition de tâche comporte un nouveau champ `credentialSpecs` pour gMSA sous Linux et Windows. Un nouveau didacticiel complet pour les gMSA sans domaine sous Windows a été ajouté, veuillez consulter Didacticiel : utilisation de conteneurs Windows avec gMSA sans domaine à l'aide de l'AWS CLI (langue française non garantie). Pour plus d'informations, consultez les sections Utilisation de gMSAs pour les conteneurs Linux et Utilisation de gMSAs pour les conteneurs Windows.	14 juillet 2023
Documentation améliorée sur les versions de l'agent ECS	La documentation relative aux versions de l'agent Amazon ECS a été mise à jour. Nous vous recommandons d'utiliser la version `v20.10.13` de Docker, ou une version plus récente, avec la dernière version de l'agent de conteneur Amazon ECS. Les versions publiées et les modifications apportées à l'agent sont disponibles sur GitHub. Pour plus d'informations, veuillez consulter Versions de l'agent de conteneur Linux Amazon ECS (langue française non garantie).	20 juin 2023
Disponibilité régionale mise à jour pour le support Fargate ARM64	La disponibilité des régions pour le support ARM64 Fargate a été mise à jour. Pour de plus amples informations, veuillez consulter Considérations.	19 juin 2023

Modification	Description	Date
Amélioration de la documentation relative à l'autoscaling de cluster	La documentation relative au dimensionnement Amazon ECS d'Amazon EC2 Auto Scaling présente des améliorations significatives en termes de précision et de lisibilité. Pour plus d'informations, veuillez consulter Autoscaling de cluster Amazon ECS (langue française non garantie).	4 mai 2023
Autorisation de balisage pour la création de ressources.	Les utilisateurs doivent être autorisés à effectuer les actions qui créent la ressource, comme `ecsCreateCluster` . Lorsque vous créez une ressource et que vous spécifiez des balises pour cette ressource, AWS effectue une autorisation supplémentaire pour vérifier qu'il existe des autorisations pour créer des balises. Pour plus d'informations, veuillez consulter Autorisation de balisage et Octroi de l'autorisation de baliser les ressources lors de la création.	18 avril 2023
Support de GMSA pour les conteneurs Linux sur EC2	Vous pouvez utiliser GMSA pour vous authentifier auprès des conteneurs Active Directory pour Linux sur. EC2 Pour plus d'informations, consultez la section Utilisation de g MSAs pour les conteneurs Linux.	14 avril 2023
Prise en charge du stockage éphémère pour les conteneurs Windows sur AWS Fargate	Vous pouvez utiliser le stockage éphémère pour les conteneurs Windows sur AWS Fargate. Pour plus d'informations, veuillez consulter Stockage des tâches Fargate (langue française non garantie).	14 avril 2023
AWS Cost Management prise en charge des données CUR au niveau des tâches	Vous pouvez activer l'utilisation des ressources et les coûts au niveau des tâches dans les rapports d'utilisation et de coût. Cela ajoute des données de répartition des coûts fractionnés pour les tâches exécutées sur AWS Fargate et. EC2 Pour de plus amples informations, veuillez consulter Rapports d'utilisation et de coût au niveau des tâches.	12 avril 2023

Modification	Description	Date
AMI Amazon Linux 2023 optimisée pour Amazon ECS	Vous pouvez déployer des charges de travail sur l'AMI Amazon Linux 2023 optimisée pour Amazon ECS. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.	10 avril 2023
AWS Fargate Norme fédérale de traitement de l'information (FIPS) 140	Vous pouvez déployer des charges de travail sur Amazon ECS conformément AWS Fargate à la norme FIPS (Federal Information Processing Standard) 140. Pour de plus amples informations, veuillez consulter AWS Fargate Norme fédérale de traitement de l'information (FIPS-140).	10 avril 2023
Suppression d'une définition de tâche	Vous pouvez supprimer une définition de tâche à l'aide de la console Amazon ECS, du SDK et de la AWS CLI. Pour plus d'informations, veuillez consulter Suppression de l'enregistrement d'une révision de définition de tâche à l'aide de la console et Définitions de tâches.	24 février 2023
AWS Fargate recommandations de service dans Compute Optimizer	AWS Compute Optimizer génère des recommandations relatives à la taille des tâches et des conteneurs en fonction de l'utilisation des tâches en cours d'exécution dans les services Amazon ECS sur AWS Fargate. Pour plus d'informations, veuillez consulter Affichage des recommandations pour les services Amazon ECS sur Fargate (langue française non garantie).	27 janvier 2023
Console Amazon ECS	La nouvelle console Amazon ECS est désormais la console par défaut. Pour plus d'informations, consultez Nouvelle console Amazon ECS.	19 janvier 2023
Politique IAM `AmazonECS_FullAccess` mise à jour	La politique IAM `AmazonECS_FullAccess` est mise à jour pour inclure des autorisations permettant d'ajouter des balises aux équilibreurs de charge lors de la création. Pour de plus amples informations, veuillez consulter Amazon ECS_ FullAccess.	4 janvier 2023

Modification	Description	Date
Utilisez des CloudWatch alarmes pour détecter les échecs de déploiement du service Amazon ECS	Vous pouvez configurer Amazon ECS pour définir le déploiement comme un échec lorsqu'il détecte qu'une CloudWatch alarme spécifiée est passée à l'état ALARM. Pour de plus amples informations, veuillez consulter the section called “Déploiements de mises à jour continues”.	19 décembre 2022
Prise en charge du mappage de ports des conteneurs	Vous pouvez définir une plage de numéros de port du conteneur liée à la plage de ports hôtes mappés dynamiquement. Pour de plus amples informations, veuillez consulter the section called “Mappages de port”.	15 décembre 2022
Disponibilité générale d'Amazon ECS Service Connect	Cette fonctionnalité ajoute la découverte de services et un maillage de services contrôlés par les déploiements du service Amazon ECS. Pour de plus amples informations, veuillez consulter the section called “ Service Connect”.	27 novembre 2022
La nouvelle expérience de la console Amazon ECS pour les définitions de tâches est mise à jour	La nouvelle expérience de la console Amazon ECS contient désormais un éditeur JSON pour les définitions de tâche. Pour de plus amples informations, veuillez consulter the section called “Création d'une définition de tâche à l'aide de la console”.	27 octobre 2022
La nouvelle expérience de la console Amazon ECS pour les définitions de tâches est mise à jour	La nouvelle expérience de la console Amazon ECS contient désormais un éditeur JSON pour les définitions de tâche. Pour de plus amples informations, veuillez consulter the section called “Création d'une définition de tâche à l'aide de la console”.	27 octobre 2022

Modification	Description	Date
La nouvelle expérience de la console Amazon ECS est mise à jour	La nouvelle expérience de la console Amazon ECS a été mise à jour avec des paramètres de service et de tâche supplémentaires. Pour plus d'informations, consultez the section called "Création d'un déploiement de mises à jour continues" et the section called "Exécution d'une application en tant que tâche".	7 octobre 2022
Nouvelles informations dans le point de terminaison des métadonnées de tâche version 4	La version 4 du point de terminaison des métadonnées de tâche inclut désormais l'identifiant VPC et le nom du service. Pour de plus amples informations, veuillez consulter the section called "Point de terminaison des métadonnées de tâches version 4".	7 octobre 2022
Tailles de la nouvelle définition de tâche	Amazon ECS sur Fargate prend désormais en charge les tailles de tâches de 8 vCPU et de 16 vCPU virtuels. Pour de plus amples informations, veuillez consulter the section called "Taille de la tâche".	16 septembre 2022
Pages CLI ECS archivées	La documentation CLI ECS a été archivée. Nous vous recommandons d'utiliser AWS Copilot pour vos besoins en matière d'outils de ligne de commande. Pour de plus amples informations, veuillez consulter Création de ressources Amazon ECS à l'aide de l'interface de ligne de commande AWS Copilot.	15 septembre 2022
Nouveaux quotas Fargate	Fargate est en train de passer des quotas basés sur le nombre de tâches aux quotas basés sur le vCPU. Pour plus d'informations, consultez la section Points de terminaison et quotas Amazon ECS.	8 septembre 2022
Support pour les pools chauds Amazon EC2 Auto Scaling.	Vous pouvez désormais utiliser les warm pools d'Amazon EC2 Auto Scaling pour étendre vos applications plus rapidement et réduire les coûts. Pour de plus amples informations, veuillez consulter Configuration d'instances pré-initialisées pour votre groupe Amazon ECS Auto Scaling.	23 mars 2022

Modification	Description	Date
Prise en charge des instances Windows dans ECS Anywhere.	ECS Anywhere prend désormais en charge les instances Windows. Pour de plus amples informations, veuillez consulter Clusters Amazon ECS pour le type de lancement externe.	3 mars 2022
Ajout de la prise en charge ECS Exec pour les instances externes	ECS Exec est désormais pris en charge pour les instances externes. Pour de plus amples informations, veuillez consulter Surveillez les conteneurs Amazon ECS avec ECS Exec.	24 janvier 2022
Mise à jour de la nouvelle expérience de la console Amazon ECS	La nouvelle expérience de console Amazon ECS prend en charge la création et la suppression d'un cluster, la mise à jour d'une définition de tâche et l'annulation de l'enregistrement d'une définition de tâche. Pour plus d'informations, consultez Création d'un cluster Amazon ECS pour le type de lancement Fargate, Suppression d'un cluster Amazon ECS, Mettre à jour une définition de tâche Amazon ECS à l'aide de la console et Annulation de l'enregistrement d'une révision de définition de tâche Amazon ECS à l'aide de la console.	8 décembre 2021
Mise à jour de la nouvelle expérience de la console Amazon ECS	La nouvelle expérience de console Amazon ECS prend en charge la création d'une définition de tâche. Pour de plus amples informations, veuillez consulter Création d'une définition de tâche Amazon ECS à l'aide de la console.	23 novembre 2021
Amazon ECS prend en charge l'architecture ARM 64 bits pour Linux.	Amazon ECS prend en charge l'architecture du processeur ARM 64 bits pour le système d'exploit ation Linux. Pour de plus amples informations, veuillez consulter the section called “Définitions de tâches pour les charges de travail ARM 64 bits”.	23 novembre 2021

Modification	Description	Date
Support d'Amazon ECS pour l'option fluentd log-driver-buffer-limit	Amazon ECS prend en charge l'option `log-driver-buffer-limit` fluentd. Pour de plus amples informations, veuillez consulter Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.	22 novembre 2021
Script de création d'AMI Linux optimisées pour Amazon ECS	Amazon ECS comporte des scripts de génération open source qui sont utilisés pour créer les variantes Linux de l'AMI optimisée pour Amazon ECS. Pour de plus amples informations, veuillez consulter Script de création d'AMI Linux optimisées pour Amazon ECS.	19 novembre 2021
État de l'instance de conteneur	Amazon ECS ajoute la prise en charge la surveillance de l'état de l'instances de conteneur. Pour de plus amples informations, veuillez consulter Surveiller l'état de l'instance de conteneur Amazon ECS.	10 novembre 2021
Prise en charge pour Windows Amazon ECS Exec	Amazon ECS Exec prend en charge Windows. Pour de plus amples informations, veuillez consulter Surveillez les conteneurs Amazon ECS avec ECS Exec.	1er novembre 2021
Prise en charge des conteneurs Windows sur Fargate.	Amazon ECS prend en charge les conteneurs Windows sur Fargate. Pour de plus amples informations, veuillez consulter Journal des modifications de version de la plateforme Windows Fargate.	28 octobre 2021
Prise en charge du GPU pour les instances externes sur Amazon ECS Anywhere	Amazon ECS prend en charge la spécification des exigences du GPU dans la définition de tâche pour les tâches exécutées sur des instances externes. Pour plus d’informations, consultez Définitions de tâches Amazon ECS pour les charges de travail du GPU et Enregistrement d'une instance externe dans un cluster Amazon ECS.	8 octobre 2021

Modification	Description	Date
Prise en charge du mode réseau `awsvpc` sous Windows	Amazon ECS prend en charge `awsvpc` sur Windows. Pour de plus amples informations, veuillez consulter Allouer une interface réseau pour une tâche Amazon ECS.	15 juillet 2021
Disponibilité générale de Bottlerocket	Une variante d'AMI optimisée pour Amazon ECS du système d'exploitation Bottlerocket est fournie sous la forme d'une AMI que vous pouvez utiliser avec Amazon ECS. Pour de plus amples informati ons, veuillez consulter Optimisé pour Amazon ECS Bottlerocket AMIs.	30 Juin 2021
Mise à jour des tâches planifiées d'Amazon ECS	Amazon EventBridge a ajouté la prise en charge de paramètres supplémentaires lors de la création de règles qui déclenchent les tâches planifiées Amazon ECS.	25 juin 2021
AWS politiques gérées pour Amazon ECS	Amazon ECS a ajouté de la documentation sur les politiques AWS gérées pour les rôles liés à un service. Pour de plus amples informations, veuillez consulter AWS politiques gérées pour Amazon Elastic Container Service.	8 juin 2021
Commencer à utiliser le AWS CDK	Ajout d'un guide de démarrage pour l'utilisation AWS CDK avec Amazon ECS. Pour de plus amples informations, veuillez consulter Création de ressources Amazon ECS à l'aide du AWS CDK.	27 mai 2021
Amazon ECS Anywhere	Amazon ECS a ajouté la prise en charge de l'enregis trement d'un serveur sur site ou d'une machine virtuelle (VM) auprès de votre cluster. Pour de plus amples informations, veuillez consulter Clusters Amazon ECS pour le type de lancement externe.	25 mai 2021

Modification	Description	Date
AMI de Windows Server 20H2 Core optimisée pour Amazon ECS	Amazon ECS a ajouté la prise en charge d'une nouvelle variante AMI optimisée pour Windows Amazon ECS pour Windows Server 20H2 Core. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.	19 avril 2021
Amazon ECS Exec	Amazon ECS a publié un nouvel outil de débogage appelé ECS Exec. Pour de plus amples informations, veuillez consulter Surveillez les conteneurs Amazon ECS avec ECS Exec.	15 mars 2021
Prise en charge de stratégie de point de terminaison d'un VPC	Amazon ECS prend désormais en charge les stratégie s de point de terminaison Amazon VPC Pour de plus amples informations, veuillez consulter Création d'une stratégie de point de terminaison d'un VPC pour Amazon ECS.	11 janvier 2021
Nouvelle expérience de console	Amazon ECS a publié une nouvelle expérience de console qui prend en charge la création ou la mise à jour d'un service ou l'exécution d'une tâche autonome. Pour plus d'informations, consultez Création d'un déploiement de mises à jour continues Amazon ECS et Exécution d'une application en tant que tâche Amazon ECS.	28 décembre 2020
Stratégie de fournisseur de capacité	Amazon ECS prend en charge la mise à jour d'un fournisseur de capacité de groupe Auto Scaling existant.	23 novembre 2020
ECS prend désormais en charge le serveur de fichiers Amazon FSx pour Windows pour les tâches Windows	Amazon ECS a ajouté la prise en charge de la spécification des volumes du serveur de fichiers Amazon FSx pour Windows dans les définitions de tâches Windows. Pour de plus amples informations, veuillez consulter Utilisation FSx pour les volumes de serveurs de fichiers Windows avec Amazon ECS.	11 novembre 2020

Modification	Description	Date
Ajout de la prise en charge du mode double pile VPC	Amazon ECS a ajouté la prise en charge de l'utilisation d'un VPC en mode double pile avec des tâches utilisant le mode `awsvpc` réseau, ce qui permet de prendre en charge les adresses. IPv6 Pour de plus amples informations, veuillez consulter Utilisation d'un VPC en mode double pile.	5 novembre 2020
Mise à jour du point de terminaison des métadonnées de tâches v4	Amazon ECS a ajouté des métadonnées supplémentaires à la sortie du point de terminaison v4 des métadonnées de tâche. Pour de plus amples informations, veuillez consulter Point de terminaison des métadonnées des tâches Amazon ECS, version 4.	5 novembre 2020
Support des Local Zones et des zones de Wavelength	Amazon ECS a ajouté la prise en charge des charges de travail dans les Local Zones et les zones Wavelength. Pour de plus amples informations, veuillez consulter Applications Amazon ECS dans des sous-réseaux partagés, des zones locales et des zones de longueur d'onde.	4 septembre 2020
Variante Amazon ECS de l'AMI Bottlerocket	Bottlerocket est un système d'exploitation open source basé sur Linux spécialement conçu pour exécuter des conteneurs. AWS Une variante d'AMI optimisée pour Amazon ECS du système d'exploitation Bottlerocket est fournie sous la forme d'une AMI que vous pouvez utiliser lors du lancement d'instances de conteneur Amazon ECS. Pour de plus amples informations, veuillez consulter Optimisé pour Amazon ECS Bottlerocket AMIs.	31 août 2020

Modification	Description	Date
Point de terminaison des métadonnées de tâches version 4 mis à jour pour les statistiques de fréquence réseau	La version 4 du point de terminaison des métadonnées des tâches a été mise à jour pour fournir des statistiq ues de débit réseau pour les tâches Amazon ECS qui utilisent le `awsvpc` ou les modes `bridge` réseau hébergés sur des EC2 instances Amazon exécutant au moins une version `1.43.0` de l'agent de conteneur . Pour de plus amples informations, veuillez consulter Point de terminaison des métadonnées des tâches Amazon ECS, version 4.	10 août 2020
Métriques d'utilisation de Fargate	AWS Fargate fournit des statistiques CloudWatch d'utilisation qui fournissent une visibilité sur l'utilisation des ressources Fargate On-Demand et Fargate Spot par vos comptes. Pour de plus amples informations, veuillez consulter Métriques d'utilisation.	3 août 2020
AWS Copilote version 0.1.0	La nouvelle AWS CLI Copilot a été lancée, fournissa nt des commandes de haut niveau pour simplifier la modélisation, la création, la publication et la gestion d'applications conteneurisées sur Amazon ECS à partir d'un environnement de développement local. Pour de plus amples informations, veuillez consulter Création de ressources Amazon ECS à l'aide de l'interface de ligne de commande AWS Copilot.	9 juillet 2020
AWS Fargate calendrie r de dépréciation des versions de plate-forme	Le calendrier de l'obsolescence des versions de la plateforme Fargate a été ajouté. Pour de plus amples informations, veuillez consulter AWS Version obsolète de la plateforme Fargate Linux.	8 juillet 2020
AWS Expansion de la région de Fargate	Amazon ECS on AWS Fargate s'est étendu à la région Europe (Milan).	25 juin 2020

Modification	Description	Date
Publication de l'AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS	Amazon ECS a publié une AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS pour les charges de travail inférentes. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.	24 juin 2020
Ajout de la prise en charge de la suppressi on des fournisseurs de capacité	Amazon ECS a ajouté la prise en charge de la suppression des fournisseurs de capacité de groupe Auto Scaling.	11 juin 2020
AWS Mise à jour de la version 1.4.0 de la plateforme Fargate	Depuis le 28 mai 2020, toute nouvelle tâche Fargate lancée à l'aide de la version 1.4.0 de la plateforme a son magasin éphémère de 20 Go chiffré avec un algorithme de chiffrement AES-256 à l'aide d'une clé de chiffrement gérée par AWS Fargate. Pour de plus amples informations, veuillez consulter Stockage éphémère des tâches Fargate pour Amazon ECS.	28 mai 2020
Prise en charge du fichier de variable d'environ nement	Ajout de la prise en charge de la spécification des fichiers de variables d'environnement dans une définition de tâche, ce qui vous permet d'ajouter en bloc des variables d'environnement à vos conteneurs. Pour de plus amples informations, veuillez consulter Transmettre une variable d'environnement individuelle à un conteneur Amazon ECS.	18 mai 2020
AWS Expansion de la région de Fargate	AWS Fargate with Amazon ECS s'est étendu à la région Afrique (Le Cap).	11 mai 2020

Modification	Description	Date
Quota de service mis à jour	Le quota de service suivant a été mis à jour : • Les clusters par compte ont été élevés de `2,000` à `10,000`. Pour plus d'informations, consultez la section Points de terminaison et quotas Amazon ECS.	17 avril 2020

Modification	Description	Date
AWS Version 1.4.0 de la plateforme Fargate	AWS La version 1.4.0 de la plateforme Fargate est publiée. Elle contient les fonctionnalités suivantes : • Ajout de la prise en charge de l'utilisation des volumes de système de fichiers Amazon EFS pour le stockage des tâches permanentes. Pour de plus amples informations, veuillez consulter Utiliser les volumes Amazon EFS avec Amazon ECS. • Le stockage des tâches éphémères est désormais de 20 Go. Pour plus d'informations, consultez Stockage éphémère des tâches Fargate pour Amazon ECS. • Le comportement du trafic réseau entrant et sortant entre les tâches a été mis à jour. À partir de la version 1.4 de la plateforme, toutes les tâches Fargate reçoivent une interface réseau Elastic unique (appelée ENI de tâche), et tout le trafic réseau passe par cette ENI au sein du VPC et vous est accessible via les journaux de vos flux VPC. Pour plus d'informations, consultez les options de mise en réseau des tâches Amazon ECS pour le type de lancement Fargate. • Tâche : ENIs ajouter un support pour les cadres jumbo. Les interfaces réseau sont configurées avec une unité de transmission maximale (MTU), qui correspond à la taille de la charge utile la plus élevée possible dans une seule trame. Plus la MTU est importante, plus la charge utile de l'application peut s'intégrer dans une seule trame. Cela réduit les frais généraux par trame et augmente l'efficac ité. La prise en charge des trames jumbo limite la	8 avril 2020

Modification	Description	Date
	surcharge lorsque le chemin réseau entre votre tâche et la destination prend en charge les trames jumbo, telles que tout le trafic qui reste dans votre VPC. • CloudWatch Container Insights inclura des mesures de performance réseau pour les tâches Fargate. Pour de plus amples informations, veuillez consulter Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée. • Ajout de la prise en charge du point de terminais on de métadonnées de tâche v4, qui fournit des informations supplémentaires pour vos tâches Fargate, y compris les statistiques réseau pour la tâche et la zone de disponibilité dans laquelle la tâche s'exécute. Pour de plus amples informati ons, veuillez consulter Point de terminaison des métadonnées des tâches Amazon ECS, version 4. • Ajout de la prise en charge du paramètre `SYS_PTRACE` Linux dans les définitions de conteneur. Pour de plus amples informations, veuillez consulter Paramètres Linux. • L'agent de conteneur Fargate remplace l'agent de conteneur Amazon ECS pour toutes les tâches Fargate. Cette modification ne devrait pas avoir d'effet sur la façon dont vos tâches s'exécutent. • L'environnement d'exécution du conteneur utilise maintenant Containerd au lieu de Docker. Cette modification ne devrait pas avoir d'effet sur la façon dont vos tâches s'exécutent. Vous remarquerez que certains messages d'erreur provenant de l'environ	

Modification	Description	Date
	nement d'exécution du conteneur mentionneront de moins en moins Docker et de en plus des erreurs générales. Pour de plus amples informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.	
Prise en charge des systèmes de fichiers Amazon EFS pour les volumes de tâches	Les systèmes de fichiers Amazon EFS peuvent être utilisés comme volumes de données pour vos tâches Amazon ECS et Fargate. Pour de plus amples informations, veuillez consulter Utiliser les volumes Amazon EFS avec Amazon ECS.	8 avril 2020
Point de terminaison des métadonnées de tâches Amazon ECS version 4	À partir de la version 1.39.0 de l'agent de conteneur Amazon ECS et de la version 1.4.0 de la plateform e Fargate, une variable d'environnement nommée `ECS_CONTAINER_METADATA_URI_V4` est injectée dans chaque conteneur d'une tâche. Lorsque vous interrogez le point de terminaison des métadonné es de tâches version 4, diverses métadonnées de tâches et statistiques Docker sont disponibles pour les tâches. Pour de plus amples informations, veuillez consulter Point de terminaison des métadonnées des tâches Amazon ECS, version 4.	8 avril 2020
Prise en charge de versions spécifiques de secrets de type Secrets Manager à injecter en tant que variables d'environnement	Ajout de la prise en charge de la spécification de données sensibles à l'aide de versions spécifiques de secrets de type Secrets Manager. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.	24 février 2020

Modification	Description	Date
Ajout d'options de configuration CodeDeploy de déploiement supplémentaires pour les déploiements bleu/vert	Le CodeDeploy service a ajouté de nouvelles configurations de déploiement Canary et linéaire pour le type de déploiement Amazon ECS. La possibilité de définir des configurations de déploiement personnalisées est également disponible. Pour de plus amples informations, veuillez consulter CodeDeploy déploiements bleu/vert pour Amazon ECS.	6 février 2020
Ajout du paramètre de définition de efsVolumeConfiguration tâche	Le paramètre de définition de tâche `efsVolumeConfiguration` est disponible en version préliminaire publique, ce qui facilite l'utilisation des systèmes de fichiers Amazon EFS avec vos tâches Amazon ECS. Pour de plus amples informations, veuillez consulter Utiliser les volumes Amazon EFS avec Amazon ECS.	17 janvier 2020
Comportement de journalisation de l'agent de conteneur Amazon ECS mis à jour	Les emplacements de journalisation de l'agent de conteneur Amazon ECS et le comportement de rotation ont été mis à jour. Pour de plus amples informations, veuillez consulter Paramètres de configuration du journal de l'agent de conteneur Amazon ECS.	13 janvier 2020
Fargate Spot	Amazon ECS a ajouté la prise en charge de l'exécution des tâches en utilisant Fargate Spot. Pour de plus amples informations, veuillez consulter Clusters Amazon ECS pour Fargate.	3 décembre 2019
Cluster Auto Scaling	L'Auto Scaling de cluster Amazon ECS vous permet de mieux contrôler la façon dont vous mettez à l'échelle les tâches au sein d'un cluster. Pour de plus amples informations, veuillez consulter Gérez automatiquement la capacité d'Amazon ECS grâce au dimensionnement automatique des clusters.	3 décembre 2019

Modification	Description	Date
Fournisseurs de capacité de cluster	Les fournisseurs de capacité de cluster Amazon ECS déterminent l'infrastructure à utiliser pour vos tâches. Pour de plus amples informations, veuillez consulter Clusters Amazon ECS.	3 décembre 2019
Création d'un cluster sur un AWS Outposts	Amazon ECS prend désormais en charge la création de clusters sur un AWS Outposts. Pour de plus amples informations, veuillez consulter the section called “Amazon Elastic Container Service sur AWS Outposts” .	3 décembre 2019
Événements d'action de service	Amazon ECS envoie désormais des événements à Amazon EventBridge lorsque certaines actions de service se produisent. Pour de plus amples informati ons, veuillez consulter Événements liés aux actions du service Amazon ECS.	25 novembre 2019
L'AMI optimisée pour le GPU Amazon ECS prend en charge les instances G4	Amazon ECS a ajouté la prise en charge de la famille de types d'instance g4 lors de l'utilisation de l'AMI optimisé pour le GPU Amazon ECS. Pour de plus amples informations, veuillez consulter Définitions de tâches Amazon ECS pour les charges de travail du GPU.	8 octobre 2019
FireLens pour Amazon ECS	FireLens pour Amazon ECS est en disponibilité générale. FireLens pour Amazon ECS vous permet d'utiliser les paramètres de définition des tâches pour acheminer les journaux vers un AWS service ou une destination partenaire à des fins de stockage et d'analyse des journaux. Pour de plus amples informati ons, veuillez consulter Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.	30 septembre 2019
AWS Expansion de la région de Fargate	AWS Fargate with Amazon ECS s'est étendu aux régions d'Europe (Paris), d'Europe (Stockholm) et du Moyen-Orient (Bahreïn).	30 septembre 2019

Modification	Description	Date
Deep Learning Containers avec Elastic Inference sur Amazon ECS	Amazon ECS prend en charge l'association d'accélérateurs Amazon Elastic Inference à vos conteneurs afin d'améliorer l'efficacité des charges de travail d'inférence de deep learning en cours d'exécution.	3 septembre 2019
FireLens pour Amazon ECS	FireLens pour Amazon ECS est en version préliminaire publique. FireLens pour Amazon ECS vous permet d'utiliser les paramètres de définition des tâches pour acheminer les journaux vers un AWS service ou une destination partenaire à des fins de stockage et d'analyse des journaux. Pour de plus amples informations, veuillez consulter Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.	30 août 2019
CloudWatch Informations sur les conteneurs	CloudWatch Container Insights est désormais disponible pour tous. Il vous permet de collecter, regrouper et récapituler les métriques et les journaux de vos applications et microservices conteneurisés. Pour de plus amples informations, veuillez consulter Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée.	30 août 2019
Configuration de l'échange au niveau du conteneur	Amazon ECS prend désormais en charge le contrôle de l'utilisation de l'espace mémoire d'échange sur vos instances de conteneur Linux au niveau du conteneur. En utilisant une configuration d'échange par conteneur, l'échange peut être activé ou désactivé pour chaque conteneur au sein d'une définition de tâche, et pour les conteneurs pour lesquels l'échange est activé, la quantité maximale d'espace d'échange utilisée peut être limitée. Pour de plus amples informations, veuillez consulter Gestion de l'espace mémoire d'échange de conteneurs sur Amazon ECS.	16 août 2019

Modification	Description	Date
AWS Expansion de la région de Fargate	AWS Fargate with Amazon ECS s'est étendu à la région Asie-Pacifique (Hong Kong).	6 août 2019
Jonction Elastic Network Interface	Ajout de types d' EC2 instances Amazon supplémen taires pris en charge pour la fonctionnalité de jonction ENI. Pour de plus amples informations, veuillez consulter Instances prises en charge pour augmenter le nombre d'interfaces réseau de conteneurs Amazon ECS.	1 août 2019
Enregistrement de plusieurs groupes cibles auprès d'un service	Ajout de la prise en charge de la spécification de plusieurs groupes cibles dans une définition de service. Pour de plus amples informations, veuillez consulter Enregistrement de plusieurs groupes cibles auprès d'un service Amazon ECS.	30 juillet 2019
Spécification de données sensibles à l'aide de secrets de Secrets Manager	Ajout d'un didacticiel pour la spécification de données sensibles à l'aide de secrets de Secrets Manager. Pour de plus amples informations, veuillez consulter Spécification de données sensibles à l'aide des secrets Secrets Manager dans Amazon ECS.	20 juillet 2019
CloudWatch Informations sur les conteneurs	Amazon ECS a ajouté la prise en charge de CloudWatch Container Insights. Pour de plus amples informations, veuillez consulter Surveillez les conteneurs Amazon ECS à l'aide de Container Insights avec une observabilité améliorée.	9 juillet 2019
Autorisations au niveau des ressources pour les services et ensembles de tâches Amazon ECS	Amazon ECS a étendu la prise en charge des autorisat ions au niveau des ressources pour les services et ensembles de tâches Amazon ECS. Pour de plus amples informations, veuillez consulter Fonctionn ement d'Amazon Elastic Container Service avec IAM.	27 juin 2019

Modification	Description	Date
Mise à jour de la nouvelle AMI optimisée pour Amazon ECS pour 2019-005 AWS	Amazon ECS a mis à jour l'AMI optimisée pour Amazon ECS afin de résoudre la vulnérabilité décrite dans AWS-2019-005.	17 juin 2019
Jonction Elastic Network Interface	Amazon ECS introduit la prise en charge du lancement d'instances de conteneur à l'aide des types d' EC2 instances Amazon pris en charge qui ont augmenté la densité de l'Elastic Network Interface (ENI). L'utilisation de ces types d'instances et du paramètre de compte `awsvpcTrunking` permet d'améliorer la densité ENI sur les instances de conteneur nouvellement lancées et de placer plus de tâches sur chaque instance de conteneur. Pour de plus amples informations, veuillez consulter Augmenter les interfaces réseau des instances de conteneur Linux Amazon ECS.	6 juin 2019
AWS Mise à jour de la version 1.3.0 de la plateforme Fargate	Depuis le 1er mai 2019, toute nouvelle tâche Fargate lancée prend en charge le pilote de journal `splunk` en plus du pilote de journal `awslogs`. Pour de plus amples informations, veuillez consulter Stockage et journalisation.	1er mai 2019
AWS Mise à jour de la version 1.3.0 de la plateforme Fargate	Depuis le 1er mai 2019, toute nouvelle tâche Fargate lancée prend en charge le référencement des données sensibles dans la configuration de journal d'un conteneur à l'aide du paramètre de définition de conteneur `secretOptions` . Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.	1er mai 2019

Modification	Description	Date
AWS Mise à jour de la version 1.3.0 de la plateforme Fargate	À compter du 2 avril 2019, toute nouvelle tâche Fargate lancée prend en charge l'injection de données sensibles dans vos conteneurs en stockant vos données sensibles dans des secrets AWS Secrets Manager AWS Systems Manager ou dans des paramètres Parameter Store, puis en les référença nt dans la définition de votre conteneur. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.	2 avril 2019
AWS Mise à jour de la version 1.3.0 de la plateforme Fargate	Depuis le 27 mars 2019, toute nouvelle tâche Fargate lancée peut utiliser des paramètres de définition de tâches supplémentaires qui vous permettent de définir une configuration proxy, des dépendances de démarrage et d'arrêt de conteneurs, ainsi qu'une valeur de temporisation d'arrêt et de démarrage par conteneur. Pour plus d’informations, consultez Configuration du proxy, Dépendances du conteneur et Temporisations de conteneurs.	27 mars 2019
Amazon ECS introduit le type de déploiement externe	Le type de déploiement externe vous permet d'utilise r n'importe quel contrôleur de déploiement tiers pour un contrôle complet du processus de déploiement pour un service Amazon ECS service. Pour de plus amples informations, veuillez consulter Déployez les services Amazon ECS à l'aide d'un contrôleur tiers.	27 mars 2019

Modification	Description	Date
AWS Deep Learning Containers sur Amazon ECS	AWS Les Deep Learning Containers sont un ensemble d'images Docker destinées à la formation et au service de modèles TensorFlow sur Amazon Elastic Container Service (Amazon ECS). Les Deep Learning Container s fournissent des environnements optimisés avec TensorFlow les bibliothèques Nvidia CUDA (pour les instances GPU) et Intel MKL (pour les instances CPU) et sont disponibles sur Amazon ECR. Pour de plus amples informations, veuillez consulter Utilisation de AWS Deep Learning Containers sur Amazon ECS.	27 mars 2019
Amazon ECS présente une gestion améliorée des dépendances de conteneurs	Amazon ECS présente des paramètres supplémen taires de définition de tâche qui vous permettent de définir des dépendances pour le démarrage et l'arrêt de conteneurs, ainsi qu'une valeur de temporisa tion d'arrêt et de lancement pour chaque conteneur. Pour de plus amples informations, veuillez consulter Dépendances du conteneur.	7 mars 2019
Amazon ECS présente l'API `PutAccoun tSettingDefault`	Amazon ECS introduit l'`PutAccountSettingD efault` API qui permet à un utilisateur de définir le statut d'adhésion ARN/ID au format par défaut pour tous les utilisateurs et rôles du compte. Auparavant, le statut d'acceptation par défaut du compte nécessitait d'utiliser le propriétaire du compte. Pour de plus amples informations, veuillez consulter Amazon Resource Names (ARNs) et IDs.	8 février 2019

Modification	Description	Date
Amazon ECS prend en charge les charges de travail GPU	Amazon ECS introduit la prise en charge des charges de travail GPU en vous permettant de créer des clusters avec des instances de conteneur GPU. Dans une définition de tâche, vous pouvez spécifier le nombre de tâches requises GPUs et l'agent ECS épinglera le physique GPUs sur le conteneur. Pour de plus amples informations, veuillez consulter Définitions de tâches Amazon ECS pour les charges de travail du GPU.	4 février 2019
Prise en charge étendue des secrets par Amazon ECS	Amazon ECS a étendu la prise en charge de l'utilisation de AWS Secrets Manager secrets directement dans vos définitions de tâches afin d'injecter des données sensibles dans vos conteneurs. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.	21 janvier 2019
Points de terminaison d'un VPC d'interface (AWS PrivateLink)	Ajout de la prise en charge pour la configuration de points de terminaison VPC optimisés par AWS PrivateLink. Cela vous permet de créer une connexion privée entre votre VPC et Amazon ECS sans avoir besoin d'un accès Internet, via une instance NAT, une connexion VPN ou AWS Direct Connect. Pour plus d'informations, veuillez consulter Points de terminaison d'un VPC d'interface (AWS PrivateLink).	26 décembre 2018

Modification	Description	Date
AWS Version 1.3.0 de la plateforme Fargate	Nouvelle version AWS de la plateforme Fargate publiée, qui contient : • Ajout de la prise en charge de l'utilisation des paramètres du AWS Systems Manager Parameter Store pour injecter des données sensibles dans vos conteneurs. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS. • Ajout du recyclage de tâche pour les tâches Fargate, qui est le processus d'actualisation des tâches qui font partie d'un service Amazon ECS service. Pour plus d'informations, consultez la section Retrait et maintenance AWS Fargate des tâches pour Amazon ECS. Pour de plus amples informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.	le 17 décembre 2018

Modification	Description	Date
Mise à jour de Service Limits	Les limites de service suivantes ont été mises à jour : • Le nombre de clusters par région et par compte a été augmenté de `1000` à `2000`. • Le nombre d'instances de conteneur par cluster a été augmenté de `1000` à `2000`. • Le nombre de services par cluster a été augmenté de `500` à `1000`. Pour de plus amples informations, veuillez consulter Quotas de service Amazon ECS service.	14 décembre 2018
AWS Expansion de la région de Fargate	AWS Fargate with Amazon ECS s'est étendu aux régions Asie-Pacifique (Mumbai) et Canada (centre). Pour de plus amples informations, veuillez consulter Régions prises en charge pour Amazon ECS sur AWS Fargate.	7 décembre 2018
blue/green Déploiements Amazon ECS	Amazon ECS a ajouté la prise en charge des blue/green déploiements à l'aide CodeDeploy de. Ce type de déploiement vous permet de vérifier un nouveau déploiement d'un service avant de lui envoyer un trafic de production. Pour de plus amples informations, veuillez consulter CodeDeploy déploiements bleu/vert pour Amazon ECS.	27 novembre 2018
Publication de l'AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS	Amazon ECS a publié un Amazon Linux 2 optimisé pour Amazon ECS AMIs pour l'architecture arm64. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.	26 novembre 2018

Modification	Description	Date
Ajout de la prise en charge d'indicateurs Docker supplémentaires dans les définitions de tâche.	Amazon ECS a introduit la prise en charge des indicateurs Docker suivants dans les définitions de tâche : • Mode IPC • Mode PID	16 novembre 2018
Prise en charge d'Amazon ECS	Amazon ECS a ajouté la prise en charge de l'utilisation des paramètres du AWS Systems Manager Parameter Store pour injecter des données sensibles dans vos conteneurs. Pour de plus amples informations, veuillez consulter Transférer des données sensibles vers un conteneur Amazon ECS.	15 novembre 2018
Identification des ressources	Amazon ECS comprend la prise en charge de l'ajout de balises de métadonnées à vos services, définitions de tâche, tâches, clusters et instances de conteneur. Pour de plus amples informations, veuillez consulter Balisage des ressources Amazon ECS.	15 novembre 2018
AWS Expansion de la région de Fargate	AWS Fargate with Amazon ECS s'est étendu aux régions de l'ouest des États-Unis (Californie du Nord) et de l'Asie-Pacifique (Séoul). Pour de plus amples informations, veuillez consulter AWS Fargatepour Amazon ECS.	7 novembre 2018

Modification	Description	Date
Mise à jour de Service Limits	Les limites de service suivantes ont été mises à jour : • Le nombre de tâches utilisant le type de lancement Fargate par région et par compte est passé de `20` à `50`. • Le nombre d'adresses IP publiques pour les tâches utilisant le type de lancement Fargate est passé de `20` à `50`. Pour de plus amples informations, veuillez consulter Quotas de service Amazon ECS service.	31 octobre 2018
AWS Expansion de la région de Fargate	AWS Fargate with Amazon ECS s'est étendu à la région Europe (Londres). Pour de plus amples informations, veuillez consulter AWS Fargatepour Amazon ECS.	26 octobre 2018
Publication d'une AMI Amazon Linux 2 optimisée pour Amazon ECS	Amazon ECS vend des systèmes Linux AMIs optimisés pour le service en deux variantes. La version la plus récente et recommandée est basée sur x;. Amazon ECS vend également des produits AMIs basés sur le, mais nous vous recommandons de migrer vos charges de travail vers la variante Amazon Linux 2, car le support de l'AMI Amazon Linux prendra fin au plus tard le 30 juin 2020. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.	18 octobre 2018

Modification	Description	Date
Point de terminaison des métadonnées de tâches Amazon ECS version 3	À partir de la version 1.21.0 de l'agent de conteneur Amazon ECS, l'agent injecte une variable d'environnement appelée `ECS_CONTAINER_METADATA_URI` dans chaque conteneur d'une tâche. Lorsque vous interrogez le point de terminaison de la version 3 des métadonnées de tâche, différentes métadonnées de tâche et statistiques Docker sont disponibles pour les tâches qui utilisent le mode réseau `awsvpc` à un point de terminaison HTTP fourni par l'agent de conteneur Amazon ECS. Pour de plus amples informations, veuillez consulter Surveillez les charges de travail à l'aide des métadonnées Amazon ECS.	18 octobre 2018
Expansion de la région de découverte de service Amazon ECS service	La prise en charge de la découverte de service Amazon ECS service a été étendue aux régions Canada (Centre), Amérique du Sud (São Paulo), Asie-Pacifique (Séoul), Asie-Pacifique (Mumbai) et Europe (Paris). Pour de plus amples informations, veuillez consulter Utilisez la découverte des services pour connecter les services Amazon ECS aux noms DNS.	le 27 septembre 2018
Ajout de la prise en charge d'indicateurs Docker supplémentaires dans les définitions de conteneur.	Amazon ECS a introduit la prise en charge des indicateurs Docker suivants dans les définitions de conteneur : • Contrôles système • Interactive • Pseudo Terminal	17 septembre 2018

Modification	Description	Date
Support de l'authent ification par registre privé pour Amazon ECS à l'aide de tâches AWS Fargate	Amazon ECS a introduit la prise en charge des tâches Fargate utilisant l'authentification de registre privé à l'aide d' AWS Secrets Manager. Cette fonction vous permet de stocker vos informations d'identification en toute sécurité, puis de les référencer dans votre définition de conteneur, ce qui permet à vos tâches d'utiliser des images privées. Pour de plus amples informations, veuillez consulter Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.	10 septembre 2018
Expansion de la région de découverte de service Amazon ECS service	La prise en charge de la découverte de service Amazon ECS service a été étendue aux régions Asie-Pacifique (Singapour), Asie-Pacifique (Sydney), Asie-Pacifique (Tokyo), UE (Francfort) et Europe (Londres). Pour de plus amples informations, veuillez consulter Utilisez la découverte des services pour connecter les services Amazon ECS aux noms DNS.	30 août 2018
Prise en charge des tâches planifiées pour les tâches Fargate	Amazon ECS a introduit la prise en charge des tâches planifiées pour le type de lancement Fargate.	28 août 2018
Authentification du registre privé à l'aide AWS Secrets Manager du support	Amazon ECS a introduit la prise en charge de l'authent ification de registre privé à l'aide d' AWS Secrets Manager. Cette fonction vous permet de stocker vos informations d'identification en toute sécurité, puis de les référencer dans votre définition de conteneur, ce qui permet à vos tâches d'utiliser des images privées. Pour de plus amples informations, veuillez consulter Utilisation d'images autres que des AWS conteneurs dans Amazon ECS.	16 août 2018

Modification	Description	Date
Ajout de la prise en charge des volumes Docker	Amazon ECS a introduit la prise en charge des volumes Docker. Pour de plus amples informations, veuillez consulter Options de stockage pour les tâches Amazon ECS.	9 août 2018
AWS Expansion de la région de Fargate	AWS Fargate with Amazon ECS s'est étendu aux régions d'Europe (Francfort), d'Asie-Pacifique (Singapour) et d'Asie-Pacifique (Sydney). Pour de plus amples informations, veuillez consulter AWS Fargatepour Amazon ECS.	19 juillet 2018

Modification	Description	Date
Ajout des stratégies de planificateur de service Amazon ECS service	Amazon ECS a introduit le concept de stratégies de planificateur de service. Deux stratégies de planificateur de service sont disponibles : • `REPLICA` – La stratégie de planification des réplicas place et gère le nombre de tâches souhaité dans votre cluster. Par défaut, le planificateur de service répartit les tâches entre les zones de disponibilité. Vous pouvez utiliser des stratégies et des contraint es de placement des tâches afin de personnaliser la façon dont les tâches sont placées. Pour de plus amples informations, veuillez consulter Stratégie de planification des répliques. • `DAEMON` – La stratégie de planification du démon déploie exactement une tâche sur chaque instance de conteneur active qui répond à toutes les contraint es de placement des tâches spécifiées dans votre cluster. Lors de l'utilisation de cette stratégie, il n'est pas nécessaire de spécifier un nombre de tâches souhaité, une stratégie de placement des tâches ou d'utiliser les politiques Service Auto Scaling. Pour de plus amples informations, veuillez consulter Stratégie de planification des démons. **Note** Les tâches Fargate ne prennent pas en charge la stratégie de planification `DAEMON`. .	12 juin 2018

Modification	Description	Date
Agent de conteneur Amazon ECS v1.18.0	Nouvelle version de l'agent de conteneur Amazon ECS, qui intègre les nouvelles fonctionnalités suivantes : • Ajout de la prise en charge de la personnalisation du comportement d'extraction d'image de l'agent de conteneur à l'aide du paramètre `ECS_IMAGE_PULL_BEHAVIOR`. Pour de plus amples informations, veuillez consulter Configuration de l'agent de conteneur Amazon ECS. Pour plus d'informations, consultez amazon-ecs-agent github.	24 mai 2018
Ajout de la prise en charge des modes réseau `bridge` et de `host` lors de la configuration de la découverte de service	Ajout de la prise en charge de la configuration de la découverte de service pour les services Amazon ECS à l'aide de définitions de tâches qui spécifient les modes réseau `bridge` ou `host`. Pour de plus amples informations, veuillez consulter Utilisez la découverte des services pour connecter les services Amazon ECS aux noms DNS.	le 22 mai 2018
Ajout de la prise en charge de paramètres supplémentaires des métadonnées de l'AMI optimisée pour Amazon ECS	Ajout de sous-paramètres qui vous permettent d'extraire par programmation l'ID, le nom d'image, le système d'exploitation, la version de l'agent de conteneur et la version d'exécution d'une AMI optimisée pour Amazon ECS. Interrogation des métadonnées via l'API Systems Manager Parameter Store. Pour de plus amples informations, veuillez consulter Récupération des métadonnées de l'AMI Linux optimisées pour Amazon ECS.	le 9 mai 2018

Modification	Description	Date
AWS Expansion de la région de Fargate	AWS Fargate with Amazon ECS s'est étendu aux régions de l'est des États-Unis (Ohio), de l'ouest des États-Unis (Oregon) et de l'ouest de l'UE (Irlande). Pour de plus amples informations, veuillez consulter AWS Fargatepour Amazon ECS.	26 avril 2018
Extraction des métadonnées d'AMI optimisée pour Amazon ECS	Ajout de la possibilité d'extraire par programmation les métadonnées d'AMI optimisée pour Amazon ECS à l'aide de l'API Systems Manager Parameter Store. Pour de plus amples informations, veuillez consulter Récupération des métadonnées de l'AMI Linux optimisées pour Amazon ECS.	10 avril 2018
AWS Version de la plateforme Fargate	Nouvelle version AWS de la plateforme Fargate publiée, qui contient : • Ajout de la prise en charge de Surveillez les charges de travail à l'aide des métadonnées Amazon ECS. • Ajout de la prise en charge de Surveillance de l'état. • Ajout de la prise en charge de Utilisez la découverte des services pour connecter les services Amazon ECS aux noms DNS Pour de plus amples informations, veuillez consulter Versions de la plateforme Fargate pour Amazon ECS.	26 mars 2018
Découverte d'Amazon ECS service	Ajout de l'intégration à Route 53 pour la prise en charge de la découverte de service Amazon ECS service. Pour de plus amples informations, veuillez consulter Utilisez la découverte des services pour connecter les services Amazon ECS aux noms DNS.	22 mars 2018

Modification	Description	Date
Prise en charge des paramètres Docker shm-size et tmpfs	Ajout de la prise en charge des paramètres Docker shm-size et tmpfs dans les définitions de tâches Amazon ECS. Pour en savoir plus sur la syntaxe mise à jour de la CLI ECS, veuillez consulter Paramètres Linux.	le 20 mars 2018
Surveillances de l'état du conteneur	Ajout de la prise en charge des surveillances de l'état Docker dans les définitions de conteneur. Pour de plus amples informations, veuillez consulter Surveillance de l'état.	8 mars 2018
AWS Fargate	Ajout d'une présentation d'Amazon ECS avec AWS Fargate. Pour de plus amples informations, veuillez consulter AWS Fargatepour Amazon ECS.	22 février 2018
Point de terminaison des métadonnées de tâches Amazon ECS	À partir de la version 1.17.0 de l'agent de conteneur Amazon ECS, différentes métadonnées de tâche et statistiques Docker sont disponibles pour les tâches qui utilisent le mode réseau `awsvpc` à un point de terminaison HTTP fourni par l'agent de conteneur Amazon ECS. Pour de plus amples informations, veuillez consulter Surveillez les charges de travail à l'aide des métadonnées Amazon ECS.	8 février 2018

Modification	Description	Date
Auto Scaling d'Amazon ECS service à l'aide des stratégies de suivi de la cible	Ajout de la prise en charge de l'Auto Scaling d'Amazon ECS service à l'aide des stratégies de suivi de la cible dans la console Amazon ECS. Pour de plus amples informations, veuillez consulter Utiliser une métrique cible pour faire évoluer les services Amazon ECS. Suppression du précédent didacticiel portant sur la mise à l'échelle par étapes dans l'assistant de première exécution d'ECS. Ce didacticiel a été remplacé par un nouveau didacticiel dédié au suivi de la cible.	8 février 2018
Prise en charge de Docker 17.09	Ajout de la prise en charge de Docker 17.09. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.	18 janvier 2018
Nouveau comportement du planificateur de service	Mise à jour des informations sur le comportement des tâches du service dont le lancement a échoué. Nouveau message d'événement de service documenté qui se déclenche lorsqu'une tâche de service connaît des échecs consécutifs.	11 janvier 2018
Période d'attente pour l'initialisation de la surveillance de l'état Elastic Load Balancing	Ajout de la possibilité de spécifier un délai d'attente pour les vérifications de l'état.	27 décembre 2017
UC et mémoire de niveau tâche	Ajout de la prise en charge de la spécification de l'UC et de la mémoire de niveau de tâche dans les définitions de tâche. Pour de plus amples informations, veuillez consulter TaskDefinition.	12 décembre 2017

Modification	Description	Date
Rôle d'exécution de tâche	L'agent de conteneur Amazon ECS effectue des appels aux actions d'API Amazon ECS à votre place. Il nécessite donc un rôle et une stratégie IAM pour que le service sache que l'agent vous appartient. Les actions suivantes sont couvertes par le rôle d'exécuti on des tâches : • Appels à Amazon ECR pour extraire l'image de conteneur • Appels pour CloudWatch stocker les journaux des applications du conteneur Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.	7 décembre 2017
Prise en charge des conteneurs Windows par GA	Ajout de la prise en charge des conteneurs Windows 2016. Pour de plus amples informations, veuillez consulter Variantes d'AMI optimisées pour Amazon ECS.	5 décembre 2017
AWS Fargate (Géorgie)	Ajout de la prise en charge du lancement des services Amazon ECS avec le type de lancement Fargate. Pour de plus amples informations, veuillez consulter Types de lancement Amazon ECS.	29 novembre 2017
Changement de nom pour Amazon ECS	Amazon Elastic Container Service est renommé (auparavant Amazon EC2 Container Service).	21 novembre 2017

Modification	Description	Date
Mise en réseau des tâches	Les fonctionnalités de mise en réseau des tâches fournies par le mode réseau awsvpc confèrent aux tâches Amazon ECS les mêmes propriétés réseau que les instances Amazon EC2 . Lorsque vous utilisez le mode réseau awsvpc dans vos définitions de tâches, chaque tâche qui est lancée à partir de cette définition de tâche obtient sa propre interface réseau Elastic, une adresse IP privée principale ainsi qu'un nom d'hôte DNS interne. La fonctionnalité de mise en réseau des tâches simplifie la mise en réseau des conteneurs et vous permet de mieux contrôler la manière dont les applications conteneurisées communiquent entre elles et avec les autres services au sein de votre entreprise. VPCs Pour de plus amples informations, veuillez consulter Options de mise en réseau des tâches Amazon ECS pour le type de EC2 lancement.	14 novembre 2017
Fichier de métadonné es de conteneur Amazon ECS	Les conteneurs Amazon ECS peuvent désormais accéder à des métadonnées telles que leur conteneur Docker ou leur ID d'image, leur configuration réseau ou Amazon ARNs. Pour de plus amples informati ons, veuillez consulter Fichier de métadonnées de conteneur Amazon ECS.	2 novembre 2017
Prise en charge de Docker 17.06	Ajout de la prise en charge de Docker 17.06. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.	2 novembre 2017
Prise en charge des indicateurs Docker : device et init	Ajout de la prise en charge des indicateurs device et init de Docker dans les définitions de tâches utilisant le paramètre `LinuxParameters` (`devices` et `initProcessEnabled`). Pour de plus amples informations, veuillez consulter LinuxParameters.	2 novembre 2017

Modification	Description	Date
Prise en charge des indicateurs Docker : cap-add et cap-drop	Ajout de la prise en charge des indicateurs cap-add et cap-drop Docker dans les définitions de tâches utilisant le paramètre `LinuxParameters` (`capabilities`). Pour de plus amples informations, veuillez consulter LinuxParameters.	22 septembre 2017
Prise en charge de Network Load Balancer	Amazon ECS a ajouté la prise en charge des équilibre urs de charge Network Load Balancers dans la console Amazon ECS.	7 septembre 2017
RunTask remplacements	Ajout de la prise en charge des remplacements de définitions de tâches lors de l'exécution d'une tâche. Cela vous permet d'exécuter une tâche tout en modifiant une définition de tâche, sans avoir à configurer une nouvelle révision de définition de tâche. Pour de plus amples informations, veuillez consulter Exécution d'une application en tant que tâche Amazon ECS.	27 juin 2017
Mise à jour des tâches planifiées d'Amazon ECS	Ajout de la prise en charge de la planification de tâches avec cron.	7 juin 2017
Instances Spot dans la console Amazon ECS	Ajout de la prise en charge de la création d'instanc es de conteneur de parc d'instances Spot dans la console Amazon ECS. Pour de plus amples informati ons, veuillez consulter Lancement d'une instance de conteneur Amazon ECS Linux.	6 juin 2017
Notification Amazon SNS pour les nouvelles versions AMI optimisées pour Amazon ECS	Ajout de la possibilité de s'abonner aux notifications SNS sur les nouvelles versions de l'AMI optimisée pour Amazon ECS.	23 mars 2017
Microservices et tâches par lots	Ajout de la documentation sur deux cas d'utilisation courants d'Amazon ECS : les microservices et les tâches par lots.	Février 2017

Modification	Description	Date
Drainage des instances de conteneur	Ajout de la prise en charge du drainage des instances de conteneur, une méthode permettant de supprimer des instances de conteneur d'un cluster. Pour de plus amples informations, veuillez consulter Vidange des instances de conteneurs Amazon ECS.	24 janvier 2017
Prise en charge de Docker 1.12	Ajout de la prise en charge de Docker 1.12. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.	24 janvier 2017
Nouvelles stratégies de placement de tâches	Ajout de la prise en charge de stratégies de placement de tâches : placement en fonction de l'attribut, BinPack, répartition par zone de disponibilité et une tâche par hôte. Pour de plus amples informations, veuillez consulter Utilisez des stratégies pour définir le placement des tâches Amazon ECS.	29 décembre 2016
Prise en charge des conteneurs Windows en version bêta	Ajout de la prise en charge des conteneurs Windows Server 2016 (version bêta). Pour de plus amples informations, veuillez consulter Variantes d'AMI optimisées pour Amazon ECS.	20 décembre 2016
Prise en charge de Blox OSS	Ajout de la prise en charge de Blox OSS, qui permet de planifier des tâches personnalisées. Pour de plus amples informations, veuillez consulter Planifiez vos conteneurs sur Amazon ECS.	1 décembre 2016
Flux d'événements Amazon ECS pour les CloudWatch événements	Amazon ECS envoie désormais les modifications de l'état des instances et des tâches du conteneur à CloudWatch Events. Pour de plus amples informati ons, veuillez consulter Automatisez les réponses aux erreurs Amazon ECS à l'aide de EventBridge.	21 novembre 2016

Modification	Description	Date
Journalisation du conteneur Amazon ECS dans CloudWatch Logs	Ajout de la prise en charge du pilote awslogs pour envoyer les flux de journaux des conteneurs à CloudWatch Logs. Pour de plus amples informations, veuillez consulter Envoyez les journaux Amazon ECS à CloudWatch .	12 septembre 2016
Services Amazon ECS avec prise en charge de l'Elastic Load Balancing pour les ports dynamiques	Ajout d'un équilibreur de charge pour prendre en charge plusieurs combinaisons instance:port par écouteur, ce qui augmente la flexibilité des conteneurs. Désormais, vous pouvez autoriser Docker à définir dynamiquement le port hôte du conteneur et le planificateur ECS enregistre la combinaison instance:port avec l'équilibreur de charge. Pour de plus amples informations, veuillez consulter Utiliser l'équilibrage de charge pour distribuer le trafic du service Amazon ECS.	11 août 2016
Rôles IAM pour les tâches Amazon ECS	Ajout de la prise en charge de l'association des rôles IAM avec une tâche. Cela fournit aux conteneurs des autorisations plus détaillées qu'un rôle unique pour une instance de conteneur complète. Pour de plus amples informations, veuillez consulter Rôle IAM de la tâche Amazon ECS.	13 juillet 2016
Prise en charge de Docker 1.11	Ajout de la prise en charge de Docker 1.11. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.	31 mai 2016
Mise à l'échelle automatique des tâches	Amazon ECS a ajouté la prise en charge de la mise à l'échelle automatique de vos tâches exécutées par un service. Pour de plus amples informations, veuillez consulter Faites évoluer automatiquement votre service Amazon ECS.	18 mai 2016

Modification	Description	Date
Filtrage des définitions de tâches par famille de tâches	Ajout de la prise en charge du filtrage d'une liste de définitions de tâches par famille de définitions de tâches. Pour de plus amples informations, veuillez consulter ListTaskDefinitions.	17 mai 2016
Journalisation des conteneurs Docker et des agents Amazon ECS	Amazon ECS a ajouté la possibilité d'envoyer les journaux des agents ECS et des conteneurs Docker depuis les instances de conteneur vers les CloudWatch journaux afin de simplifier la résolution des problèmes.	5 mai 2016
Prise en charge d'Amazon Linux 2016.03 par l'AMI optimisée pour ECS.	L'AMI optimisée pour ECS a ajouté la prise en charge d'Amazon Linux 2016.03. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.	5 avril 2016
Prise en charge de Docker 1.9	Ajout de la prise en charge de Docker 1.9. Pour de plus amples informations, veuillez consulter Linux optimisé pour Amazon ECS AMIs.	22 décembre 2015
CloudWatch métriques pour la réservation du processeur et de la mémoire du cluster	Amazon ECS a ajouté des CloudWatch métriques personnalisées pour la réservation du processeur et de la mémoire.	22 décembre 2015
Nouvelle expérienc e de mise en route Amazon ECS	Ajout de la création de rôles zéro clic lors de la première exécution de la console Amazon ECS.	23 novembre 2015
Placement des tâches dans les zones de disponibilité	Ajout de la prise en charge du placement des tâches dans les zones de disponibilité par le planificateur de services Amazon ECS service.	8 octobre 2015

Modification	Description	Date
CloudWatch métriques pour les clusters et services Amazon ECS	Amazon ECS a ajouté CloudWatch des métriques personnalisées pour l'utilisation du processeur et de la mémoire pour chaque instance de conteneur , service et famille de définition de tâche dans un cluster. Ces nouvelles métriques peuvent être utilisées pour dimensionner les instances de conteneur dans un cluster à l'aide de groupes Auto Scaling ou pour créer des CloudWatch alarmes personnalisées.	17 août 2015
Prise en charge des ports UDP	Ajout de la prise en charge des ports UDP dans les définitions de tâches.	7 juillet 2015
Remplacements de variable d'environnement	Ajout de la prise en charge deregisterTaskDefinition et du remplacement des variables d'environnement pour RunTask.	18 juin 2015
Automatisation des mises à jour de l'agent Amazon ECS	Ajout de la possibilité de voir la version de l'agent ECS en cours d'exécution sur une instance de conteneur . Également capable de mettre à jour l'agent ECS à partir du SDK AWS Management Console AWS CLI, et.	11 juin 2015
Planificateur de service Amazon ECS service et intégration d'Elastic Load Balancing	Ajout de la possibilité de définir un service et de l'associer à un équilibreur de charge Elastic Load Balancing.	9 avril 2015
Disponibilité générale d'Amazon ECS	Disponibilité générale d'Amazon ECS dans les régions USA Est (Virginie du Nord), USA Ouest (Oregon), Asie-Pacifique (Tokyo) et UE (Irlande).	9 avril 2015

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.