aws

Guide du développeur

Amazon Elastic Container Service

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon Elastic Container Service: Guide du développeur

Copyright © 2024 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 Amazon ECS 1
Capacité Amazon ECS 2
Contrôleur Amazon ECS 3
Provisionnement Amazon ECS 3
Cycle de vie des applications 3
Informations connexes 5
Premiers pas 8
Configuration 8
Inscrivez-vous pour un Compte AWS 8
Création d'un utilisateur administratif 9
Créer un Virtual Private Cloud 10
Création d’un groupe de sécurité 11
Créer les informations d'identification pour vous connecter à votre instance EC2 16
Installez le AWS CLI 16
Création d'une image de conteneur 17
Prérequis 17
Créer une image Docker 19
Transmettre votre image à Amazon Elastic Container Registry 21
Nettoyage 23
Étapes suivantes 23
Utilisation de la console avec des conteneurs Linux sur AWS Fargate 23
Prérequis 24
Étape 1 : créer le cluster 24
Étape 2 : Créer une définition de tâche 25
Étape 3 : Créer le service 27
Étape 4 : Afficher votre service 27
Étape 5 : nettoyer 27
Commencer à utiliser Windows sur Amazon EC2 28
Prérequis 28
Étape 1 : créer un cluster 29
Étape 2 : Enregistrer une définition de tâche 31
Étape 3 : Créer un service 32
Étape 4 : Afficher votre service 33

Étape 5 : Nettoyer 34
Présentation des outils pour développeur 35
AWS Management Console 35
AWS Command Line Interface 36
AWS CloudFormation 36
CLI AWS Copilot 37
AWS CDK 37
AWS App2Container 38
CLI Amazon ECS 38
Intégration de Docker Desktop avec Amazon ECS 39
Kits SDK AWS 39
Récapitulatif 40
Utilisation de la CLI AWS Copilot 41
Installation de la CLI AWS Copilot 41
Mise en route avec AWS Copilot 50
Prérequis 50
Déployer votre application à l'aide d'une commande 51
Déployer votre application étape par étape 51
Utilisation du AWS CDK 57
Étape 1 : Configurer votre projet AWS CDK 58
Étape 2 : Utiliser AWS CDK pour définir un serveur web conteneurisé sur Fargate 60
Étape 3 : Tester le service web 65
Étape 4 : Nettoyer 66
Étapes suivantes 66
Création de ressources Amazon ECS avec AWS CloudFormation 67
Amazon ECS et modèles AWS CloudFormation 67
Exemple de modèles 68
Utilisation de AWS CLI pour créer des ressources à partir de modèles 75
En savoir plus sur AWS CloudFormation 75
Utilisation de la CLI Amazon ECS 76
Installation de la CLI Amazon ECS 76
Configuration de la CLI Amazon ECS 85
AWS Fargate 88
Procédures pas à pas de Fargate 88
Fournisseurs de capacité 89
Définitions de tâche 89

Versions de plateforme ... 89
Équilibrage de charge des services ... 90
Métriques d'utilisation ... 90
Versions de plateforme Linux Fargate ... 91
Considérations ... 91
1.4.0 ... 92
1.3.0 ... 94
Migration vers la version 1.4.0 de la plateforme Linux ... 95
Obsolescence de la version de plateforme ... 96
Versions de plateformes Windows Fargate ... 99
Remarques relatives à la version de plateforme ... 99
1.0.0 ... 100
Considérations sur les conteneurs Windows sur Fargate ... 100
AWS FAQ sur la maintenance des tâches Fargate ... 101
Qu'est-ce que le maintien des tâches et la mise hors service de Fargate ? ... 101
Que contient l'avis de mise hors service des tâches ? ... 102
Puis-je modifier le temps d'attente pour la retraite des tâches ? ... 105
Puis-je recevoir des notifications de retrait de tâches par le biais d'autres AWS services ? ... 106
Puis-je modifier le retrait d'une tâche une fois qu'il a été planifié ? ... 106
Puis-je contrôler le calendrier de remplacement d'une tâche ? ... 106
Comment Amazon ECS gère-t-il les tâches faisant partie d'un service ? ... 106
Amazon ECS peut-il gérer automatiquement des tâches autonomes ? ... 107
AWS Régions de Fargate ... 107
Conteneurs Linux sur AWS Fargate ... 107
Conteneurs Windows sur AWS Fargate ... 109
Conception de votre solution pour Amazon ECS ... 112
Capacité Amazon ECS ... 112
Réseaux ... 112
Accès aux fonctionnalités ... 113
Journalisation ... 114
Types de lancement ... 115
Type de lancement Fargate ... 115
Type de lancement EC2 ... 117
Type de lancement externe ... 118
Utilisation de sous-réseaux partagés, de zones locales et de zones de longueur d'onde ... 118
Sous-réseaux partagés ... 119

Zones locales 120
Zones Wavelength 121
Amazon Elastic Container Service sur AWS Outposts 122
Prérequis 122
Limites 122
Considérations relatives à la connectivité réseau 122
Création d'un cluster Amazon ECS sur un AWS Outposts 123
Bonnes pratiques pour connecter les applications Amazon ECS à Internet 126
Sous-réseau public et passerelle Internet 127
Sous-réseau privé et passerelle NAT 129
Bonnes pratiques pour recevoir des connexions entrantes vers Amazon ECS depuis Internet .. 130
Application Load Balancer 131
Network Load Balancer 132
API HTTP Amazon API Gateway 134
Accès aux fonctionnalités d'Amazon ECS via les paramètres du compte 136
Amazon Resource Names (ARN) et ID 141
Calendrier relatif au format ARN et de l'ID de ressource 143
AWS Fargate Conformité à la norme fédérale de traitement de l'information (FIPS-140) 143
Autorisation de balisage 144
Chronologie des autorisations de balisage 145
AWS Fargate temps d'attente pour la retraite des tâches 146
Surveillance du temps d'exécution (GuardDuty intégration Amazon) 147
Affichage de vos paramètres de compte à l'aide de la console 148
Modification des paramètres de compte 148
Restauration des paramètres par défaut du compte Amazon ECS 149
Gestion des paramètres de compte à l'aide d’ AWS CLI 149
Définitions de tâches 152
États de définition de tâche 153
Ressources Amazon ECS pouvant bloquer une suppression 154
Architecture de votre application 155
Bonnes pratiques pour les images de conteneurs 157
Bonnes pratiques relatives à la taille des tâches Amazon ECS 159
Mise en réseau des tâches pour les tâches sur les instances Amazon EC2 160
Mise en réseau des tâches pour les tâches sur Fargate 173
Stockage éphémère de la tâche Fargate 177
Utilisation de volumes de données dans des tâches 179

Gestion de l'espace d'échange de conteneur 229
Considérations relatives à Fargate 231
Considérations relatives à EC2 Windows 240
Création d'une définition de tâche à l'aide de la console 241
Validation JSON 242
AWS CloudFormation piles 242
Procédure 242
Mise à jour d'une définition de tâche à l'aide de la console 273
Validation JSON 274
Annulation de l'enregistrement d'une révision de définition de tâche à l'aide de la console 275
AWS CloudFormation piles 242
Suppression de l'enregistrement d'une révision de définition de tâche à l'aide de la console 276
Ressources Amazon ECS pouvant bloquer une suppression 154
Cas d'utilisation de définition de tâche 278
Utilisation des GPU sur Amazon ECS 279
Utilisation du transcodage vidéo sur Amazon ECS 288
Utilisation de AWS Neuron sur Amazon Linux 2 sur Amazon ECS 302
Utilisation d'instances DL1 Deep Learning sur Amazon ECS 310
Utilisation des charges de travail ARM 64 bits sur Amazon ECS 312
Utilisation du pilote du journal awslogs 315
Utilisation du routage personnalisé des journaux 323
Authentification de registre privé pour les tâches 346
Utiliser les paramètres de définition des tâches pour transmettre des variables d'environnement à un conteneur 349
Utiliser un fichier pour transmettre des variables d'environnement à un conteneur 350
Transmission de données sensibles à un conteneur 354
Exemples de définitions de tâches 376
serveur Web 376
`splunk`pilote de journal 378
`fluentd`pilote de journal 379
`gelf`pilote de journal 380
Charges de travail sur les instances externes 380
Image Amazon ECR et définition des tâches (rôle IAM) 382
Point d'entrée avec commande 382
Dépendances du conteneur 383
Exemples de définitions de tâche Windows 385

Clusters et capacité 387
Concepts des fournisseurs de capacité 387
Fournisseurs de capacité 389
Types de fournisseurs de capacité 389
Considérations relatives aux fournisseurs de capacité 390
AWS Fargate fournisseurs de capacité 391
Fournisseurs de capacité de groupe Amazon EC2 Auto Scaling 394
Cluster Auto Scaling 397
Considérations 399
Présentation de la mise à l'échelle automatique des clusters 400
Protection contre la résiliation gérée 402
Mise à jour de la façon dont Amazon ECS crée des ressources de mise à l'échelle automatique d'un cluster 409
Activation de l'autoscaling de cluster 410
Désactivation du cluster Auto Scaling 411
Gestion du cluster 413
Création d'un cluster pour le type de lancement Fargate et External à l'aide de la console 415
Création d'un cluster pour le type de lancement Amazon EC2 à l'aide de la console 417
Mise à jour d'un cluster à l'aide de la console 422
Suppression d'un cluster avec la console 424
Création d'un fournisseur de capacité pour un cluster à l'aide de la console 425
Mise à jour d'un fournisseur de capacité pour un cluster à l'aide de la console 426
Suppression d'un fournisseur de capacité pour un cluster à l'aide de la console 426
Création de capacité 427
Spot instances 429
Instances Linux 433
instances Windows 468
Instances externes 511
Gestion de capacité 534
Versions d'agent de conteneur 535
Gestion des instances de conteneur Linux 550
Gestion des instances de conteneur Windows 626
Drainage d'instance géré par Amazon ECS 645
Planification de vos conteneurs 659
Planificateur de service 659
Tâches autonomes 661

Planificateurs personnalisés 661
Cycle de vie des tâches Amazon ECS 661
États de cycle de vie 662
Comment Amazon ECS place les tâches sur les instances de conteneur 664
Type de lancement EC2 664
Type de lancement Fargate 665
Utiliser des stratégies pour définir le placement des tâches 665
Tâches liées au groupe 671
Définissez les instances de conteneur utilisées pour les tâches 672
Tâches autonomes 683
Flux de travail des tâches 683
Bonnes pratiques pour améliorer le temps de lancement des tâches 684
Exécuter une application en tant que tâche Amazon ECS 686
Exécutez des tâches Amazon ECS selon des plannings à l'aide d'Amazon EventBridge Scheduler 698
Arrêter une tâche Amazon ECS 705
Services 706
Concepts du planificateur de service 706
Concepts du service supplémentaire 711
Bonnes pratiques pour améliorer les délais de déploiement 712
Création d'un service 723
Mise à jour d'un service 756
Mettre à jour une configuration de déploiement bleu/vert 776
Suppression d'un service 777
Types de déploiement 778
Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge 801
Faites évoluer automatiquement votre service Amazon ECS 811
Interconnexion des services 825
Protégez vos tâches Amazon ECS pour éviter qu'elles ne soient interrompues par des événements évolutifs 871
Logique de régulation du service Amazon ECS 879
Balisage des ressources Amazon ECS 882
Comment les ressources sont étiquetées 883
Balisage des ressources à la création 886
Restrictions 886
Balises gérées par Amazon ECS 887

Identification de vos ressources pour facturation 888
Gestion des étiquettes à l'aide de la console 889
Ajout de balises sur une ressource individuelle lors du lancement 889
Gestion des balises de ressources individuelles 890
Ajout de balises à une instance de conteneur Amazon EC2 891
Marquage d'une instance de conteneur externe 893
Gestion des balises à l'aide de la CLI ou de l'API 893
Rapports d'utilisation 895
Utilisation et coût au niveau des tâches 896
Surveillance 899
Bonnes pratiques pour la surveillance d'Amazon ECS 900
Outils de surveillance pour Amazon ECS 901
Outils automatiques 901
Outils manuels 903
Surveillez Amazon ECS à l'aide de CloudWatch 904
Considérations 904
Mesures et dimensions disponibles pour Amazon ECS 905
AWS Fargate métriques d'utilisation 924
Affichage des métriques Amazon ECS 925
Automatisez les réponses aux erreurs Amazon ECS à l'aide de EventBridge 927
Événements Amazon ECS 928
Gestion des événements 947
Surveillez les conteneurs Amazon ECS à l'aide de Container Insights 951
Considérations 951
Configuration de CloudWatch Container Insights pour Amazon ECS 952
Autorisations requises pour que CloudWatch Container Insights puisse consulter les événements du cycle de vie d'Amazon ECS 953
Déterminer l'état des tâches à l'aide de vérifications de l'état des conteneurs 955
Comment Amazon ECS détermine l'état de santé des tâches 957
Afficher l'état du conteneur 958
Surveiller l'état de l'instance de conteneur Amazon ECS 958
Rubriques en relation 959
Identifiez les opportunités d'optimisation d'Amazon ECS à l'aide des données de suivi des applications 959
Autorisations IAM requises pour AWS Distro pour OpenTelemetry l'intégration avec AWS X-Ray 960

Spécification de la AWS distribution pour le OpenTelemetry sidecar à AWS X-Ray intégrer dans la définition de votre tâche ... 962
Corrélez les performances des applications Amazon ECS à l'aide des métriques des applications ... 963
Exporter les métriques des applications vers Amazon CloudWatch ... 964
Exportation des métriques d'application vers Amazon Managed Service for Prometheus ... 968
Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail ... 973
Informations Amazon ECS dans CloudTrail ... 973
Présentation des entrées des fichiers journaux Amazon ECS ... 974
Identifiez les comportements non autorisés grâce à la surveillance du temps d'exécution ... 976
Comment fonctionne la surveillance du temps d'exécution avec Amazon ECS ... 977
Considérations ... 978
Utilisation des ressources ... 978
GuardDuty gestion des agents ... 979
Gestion manuelle de la surveillance du temps d'exécution ... 982
Questions fréquentes sur le dépannage ... 986
Surveillez les conteneurs Amazon ECS avec ECS Exec ... 991
Considérations relatives à l'utilisation d'ECS Exec ... 991
Conditions préalables à l'utilisation d'ECS Exec ... 994
Architecture ... 994
Utilisation d'ECS Exec ... 995
Journalisation et audit à l'aide d'ECS Exec ... 997
Utilisation de stratégies IAM pour limiter l'accès à ECS Exec ... 1002
Recommandations Compute Optimizer ... 1005
Recommandations relatives à la taille des tâches pour Fargate ... 1005
Sécurité ... 1007
Gestion des identités et des accès ... 1008
Public ciblé ... 1009
Authentification par des identités ... 1009
Gestion des accès à l’aide de politiques ... 1013
Fonctionnement d'Amazon Elastic Container Service avec IAM ... 1016
Exemples de politiques basées sur l’identité ... 1028
AWS politiques gérées pour Amazon ECS ... 1041
Utilisation des rôles liés à un service ... 1074
Autorisations requises pour la console Amazon ECS ... 1078
Rôle IAM d'exécution de tâche ... 1085

Rôle IAM de tâche 1092
Rôle IAM dans l'infrastructure 1104
Configuration supplémentaire d'un rôle pour les tâches Windows 1111
Rôle IAM d'instance de conteneur 1113
Rôle IAM ECS Anywhere 1120
CodeDeploy Rôle IAM 1122
EventBridge Rôle IAM 1129
Autorisations IAM requises pour Service Auto Scaling 1133
Baliser des ressources pendant la création 1135
Résolution des problèmes 1139
Journalisation et surveillance 1142
Validation de conformité 1144
AWS Fargate Conformité à la norme FIPS-140 1145
Considérations 1145
Utilisation de FIPS sur Fargate 1146
Utilisation à CloudTrail des fins d'audit 1146
Sécurité de l’infrastructure 1148
Points de terminaison d’un VPC d'interface (AWS PrivateLink) 1148
Bonnes pratiques 1153
AWS Identity and Access Management 1154
Utilisation de rôles IAM avec les tâches Amazon ECS 1157
Sécurité du réseau 1165
Gestion des secrets 1170
Utilisation d'informations d'identification de sécurité temporaires avec des opérations d'API 1173
Conformité et sécurité 1173
Journalisation et surveillance 1175
AWS Fargate sécurité 1177
Sécurité d'instance de conteneur EC2 1180
Sécurité des tâches et des conteneurs 1181
Sécurité de l'exécution 1188
Bonnes pratiques en matière d'AMI 1189
AWS Les partenaires 1190
Récupérez les métadonnées Amazon ECS 1191
Fichier de métadonnées de conteneur 1192
Emplacements des fichiers de métadonnées de conteneur 1192

Activation des métadonnées de conteneur 1193
Format des fichiers de métadonnées de conteneur 1194
Métadonnées de tâches disponibles pour les tâches Amazon ECS sur EC2 1198
Point de terminaison des métadonnées de tâches version 4 1199
Point de terminaison des métadonnées de tâches version 3 1225
Point de terminaison des métadonnées de tâches version 2 1233
Métadonnées de tâches disponibles pour les tâches sur Fargate 1241
Point de terminaison des métadonnées de tâches version 4 pour les tâches sur Fargate ... 1241
Point de terminaison des métadonnées de tâches Fargate v3 1258
Introspection des conteneurs 1264
Utilisation avec d'autres services 1268
Utilisation d'images Amazon ECR avec Amazon ECS 1268
Utilisation d'images Amazon ECR avec Amazon ECS 1268
AWS Deep Learning Containers sur Amazon ECS 1269
Les conteneurs Deep Learning Containers avec Elastic Inference sur Amazon ECS 1270
Utilisation Notifications des utilisateurs AWS avec Amazon ECS 1271
Exemple 1271
Didacticiels 1272
Utilisation de conteneurs Windows sur AWS Fargate 1272
Prérequis 1273
Étape 1 : créer un cluster 1273
Étape 2 : Enregistrer une définition de tâche Windows 1274
Étape 3 : Créer un service avec votre définition de tâche 1276
Étape 4 : Afficher votre service 1276
Étape 5 : Nettoyer 1277
Création d'un cluster avec une tâche Fargate Linux à l'aide du AWS CLI 1278
Prérequis 1278
Étape 1 : Créer un cluster 1279
Étape 2 : Enregistrer une définition de tâche Linux 1280
Étape 3 : Répertorier les définitions de tâche 1281
Étape 4 : Créer un service 1282
Étape 5 : Répertorier les services 1282
Étape 6 : Décrire le service en cours d'exécution 1283
Étape 7 : Test 1285
Étape 8 : Nettoyer 1289
Création d'un cluster avec une tâche Fargate Windows à l'aide du AWS CLI 1289

Prérequis 1290
Étape 1 : Créer un cluster 1291
Étape 2 : Enregistrement d'une définition de tâche Windows 1291
Étape 3 : Répertorier les définitions de tâche 1293
Étape 4 : Créer un service 1293
Étape 5 : Répertorier les services 1294
Étape 6 : Décrire le service en cours d'exécution 1294
Étape 7 : Nettoyer 1296
Création d'un cluster avec une tâche EC2 à l'aide du AWS CLI 1297
Prérequis 1297
Étape 1 : Créer un cluster 1298
Étape 2 : Lancer une instance avec l'AMI Amazon ECS 1299
Étape 3 : Répertorier les instances de conteneur 1299
Étape 4 : Décrire votre instance de conteneur 1299
Étape 5 : Enregistrer une définition de tâche 1302
Étape 6 : Répertorier les définitions de tâche 1304
Étape 7 : Exécuter une tâche 1304
Étape 8 : Répertorier les tâches 1306
Étape 9 : Décrire la tâche en cours d'exécution 1306
Utilisation de la mise à l'échelle automatique du cluster avec la console AWS Management Console et la console Amazon ECS 1307
Prérequis 1307
Étape 1 : Créer un cluster Amazon ECS 1308
Étape 2 : Enregistrer une définition de tâche 1309
Étape 3 : Exécuter une tâche 1310
Étape 4 : Vérifier 1310
Étape 5 : nettoyer 1312
Spécification de données sensibles à l'aide de secrets de Secrets Manager 1313
Prérequis 1313
Étape 1 : Créer un secret Secrets Manager 1313
Étape 2 : Mettre à jour votre rôle IAM d'exécution de tâche 1313
Étape 3 : Créer une définition de tâche Amazon ECS 1315
Étape 4 : Créer un cluster Amazon ECS 1316
Étape 5 : Exécuter une tâche Amazon ECS 1316
Étape 6 : Vérification 1317
Étape 7 : Nettoyer 1318

Création d'un service à l'aide de Service Discovery 1319
Prérequis 1319
Étape 1 : créer les ressources Service Discovery dans AWS Cloud Map 1319
Étape 2 : Créer les ressources Amazon ECS 1321
Étape 3 : vérifier la découverte du service dans AWS Cloud Map 1324
Étape 4 : Nettoyer 1326
Création d'un service à l'aide d'un déploiement bleu/vert 1328
Prérequis 1328
Étape 1 : Créer un Application Load Balancer 1329
Étape 2 : créer un cluster Amazon ECS 1330
Étape 3 : Enregistrer une définition de tâche 1331
Étape 4 : créer un service Amazon ECS service 1332
Étape 5 : Créer les ressources AWS CodeDeploy 1333
Étape 6 : Création et surveillance d'un CodeDeploy déploiement 1336
Étape 7 : nettoyer 1342
Écouter les CloudWatch événements Amazon ECS 1344
Prérequis : Configuration d'un cluster test 1344
Étape 1 : Créer la fonction Lambda 1344
Étape 2 : Enregistrer une règle d'événement 1345
Étape 3 : Créer une définition de tâche 1346
Étape 4 : Test de la règle 1347
Envoi d'alertes Amazon Simple Notification Service en cas d'arrêt d'une tâche 1347
Prérequis : Configuration d'un cluster test 1348
Prérequis : configurer les autorisations pour Amazon SNS 1348
Étape 1 : Créer une rubrique Amazon SNS et s'y abonner 1348
Étape 2 : Enregistrer une règle d'événement 1349
Étape 3 : Test de la règle 1350
Concaténation des messages du journal multiligne ou de suivi de pile 1351
Autorisations IAM requises 1352
Déterminez à quel moment utiliser le paramètre de journal multiligne 1353
Options d'analyse et de concaténation 1355
Didacticiel : Utilisation d'Amazon EFS 1374
Étape 1 : Créer un cluster Amazon ECS 1375
Étape 2 : créer un groupe de sécurité pour des instances Amazon EC2 et le système de fichiers Amazon EFS 1376
Étape 3 : Créer un système de fichiers Amazon EFS 1377

Étape 4 : Ajouter du contenu au système de fichiers Amazon EFS 1378
Étape 5 : Créer une définition de tâche 1380
Étape 6 : Exécuter une tâche et afficher les résultats 1382
Didacticiel : Utilisation de FSx for Windows File Server 1384
Prérequis pour le didacticiel 1385
Étape 1 : Créer des rôles IAM d'accès 1386
Étape 2 : Créer un Active Directory (AD) Windows 1387
Étape 3 : Vérifier et mettre à jour votre groupe de sécurité 1387
Étape 4 :Créer un système de fichiers Amazon FSx for Windows File Server 1389
Étape 5 : créer un cluster Amazon ECS 1390
Étape 6 : créer une instance Amazon EC2 optimisée pour Amazon ECS 1391
Étape 7 : Enregistrement d'une définition de tâche Windows 1392
Étape 8 : Exécuter une tâche et afficher les résultats 1395
Étape 9 : nettoyer 1396
Déploiement de Fluent Bit sur des conteneurs Amazon ECS pour Windows 1397
Prérequis 1400
Étape 1 : Créer les rôles IAM d'accès 1400
Étape 2 : Créer une instance de conteneur Windows Amazon ECS 1401
Étape 3 : Configurer Fluent Bit 1403
Étape 4 : enregistrer une définition de tâche Windows Fluent Bit qui achemine les journaux vers CloudWatch 1405
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 1407
Étape 6 : Enregistrer une définition de tâche Windows qui génère les journaux 1408
Étape 7 : Créer la définition de tâche `windows-app-task` 1409
Étape 8 : Vérifiez les connexions CloudWatch 1410
Étape 9 : Nettoyer 1411
Utilisation des GMSA pour les conteneurs Windows sur Amazon EC2 1412
Considérations 1413
Prérequis 1414
Configuration 1415
Utilisation de conteneurs Windows avec Domainless à gMSA l'aide du AWS CLI 1421
Prérequis 1422
Étape 1 : créer et configurer le compte gMSA sur les services de domaine Active Directory (AD DS) 1423
Étape 2 : charger les informations d'identification dans Secrets Manager 1425

Étape 3 : modifier votre JSON CredSpec pour inclure des informations gMSA sans domaine ... 1426
Étape 4 : charger CredSpec dans Amazon S3 ... 1427
Étape 5 (facultative) : créer un cluster Amazon ECS ... 1428
Étape 6 : créer un rôle IAM pour les instances de conteneur ... 1429
Étape 7 : créer un rôle d'exécution de tâche personnalisé ... 1429
Étape 8 : créer un rôle de tâche pour Amazon ECS Exec ... 1430
Étape 9 : enregistrer une définition de tâche ... 1432
Étape 10 : enregistrer une instance de conteneur Windows ... 1433
Étape 11 : vérifier l'instance de conteneur ... 1434
Étape 12 : exécuter une tâche Windows ... 1435
Étape 13 : vérifier que le conteneur possède les informations d'identification gMSA ... 1436
Étape 14 : nettoyer ... 1437
Débogage ... 1438
Utilisation gMSA pour les Linux conteneurs sur Amazon EC2 ... 1439
Considérations ... 1440
Prérequis ... 1441
Configuration ... 1442
CredSpec file ... 1449
Utilisation gMSA pour les Linux conteneurs sur Fargate ... 1450
Considérations ... 1451
Prérequis ... 1451
Configuration ... 1452
CredSpec file ... 1455
Utilisation d'EC2 Image Builder pour créer des AMI personnalisées optimisées pour Amazon ECS ... 1456
Utilisation de l'ARN de l'image avec l'infrastructure en tant que code (IaC) ... 1458
Utilisation de l'ARN de l'image avec AWS CloudFormation ... 1460
Utilisation de l'ARN de l'image avec Terraform ... 1462
Résolution des problèmes ... 1463
Résolution des problèmes avec ECS Exec ... 1463
Vérifier à l'aide d'Amazon ECS Exec Checker ... 1464
Erreur lors de l'appel à `execute-command` ... 1464
Résolution des problèmes liés à Amazon ECS Anywhere ... 1464
Problèmes d'enregistrement d'instance externe ... 1465
Problèmes de réseau d'instance externe ... 1466

Problèmes d'exécution de tâches 1466
Vérification des erreurs de tâches arrêtées 1466
Ressources supplémentaires 1468
Codes d'erreur des tâches arrêtées 1469
CannotPullContainer erreurs de tâche 1472
Messages d'événement de service 1477
Messages d'événement de service 1478
Valeur d'UC ou de mémoire non valide 1488
`CannotCreateContainerError: API error (500): devmapper` 1490
Résolution des problèmes liés aux équilibreurs de charge des services 1492
Résolution des problèmes de connexion à un volume Amazon EBS 1494
Vérification des causes de défaillance de la connexion au volume 1494
Scénarios d'échec de l'attachement à un volume Amazon EBS 1495
Dépannage de la scalabilité automatique de service 1500
Utiliser la sortie de débogage Docker 1500
Emplacements des fichiers journaux Amazon ECS 1502
Journal de l'agent de conteneur Amazon ECS 1502
Journal `ecs-init` Amazon ECS 1506
Journal d'audit des informations d'identification des rôles IAM pour les tâches 1506
Collecteur de journaux Amazon ECS 1507
Diagnostics d'introspection d'agent 1509
Diagnostics Docker 1512
Dresser une liste des conteneurs Docker 1512
Afficher les journaux Docker 1513
Inspecter les conteneurs Docker 1514
AWS Fargate limitation des quotas 1515
Limitation de l'API `RunTask` 1516
Ajustement des quotas de taux 1517
Motifs d'échec d'API 1517
Bonnes pratiques pour gérer les problèmes de régulation 1528
Régulation synchrone 1528
Régulation asynchrone 1528
Surveillance de l'étranglement 1529
Utilisation CloudWatch pour surveiller l'étranglement 1531
Références de paramètres et modèles de ressources 1532
Paramètres de définition de tâche 1532

Famille ... 1532
Types de lancement ... 1533
Rôle de tâche ... 1533
Rôle d'exécution de tâche ... 1534
Mode réseau ... 1534
Plateforme d'exécution ... 1536
Taille de la tâche ... 1537
Définitions de conteneur ... 1541
Nom de l'accélérateur Elastic Inference ... 1589
Contraintes de placement des tâches ... 1590
Configuration du proxy ... 1591
Volumes ... 1593
Balises ... 1601
Autres paramètres de définition de tâche ... 1602
Modèle de définition de tâche ... 1604
Paramètres de définition de service ... 1616
Type de lancement ... 1616
Stratégie de fournisseur de capacité ... 1616
Définition de tâche ... 1618
Système d'exploitation de la plateforme ... 1618
Version de plateforme ... 1619
Cluster ... 1620
Nom du service ... 1620
Stratégie de planification ... 1620
Nombre souhaité ... 1621
Configuration de déploiement ... 1621
Contrôleur de déploiement ... 1624
Placement des tâches ... 1625
Étiquettes ... 1627
Configuration réseau ... 1628
Jeton client ... 1637
Configurations de volume ... 1637
Modèle de définition de service ... 1643
Service Quotas ... 1650
Service Quotas Amazon ECS ... 1650
AWS Fargate quotas de service ... 1654

Gérer vos quotas de service dans le AWS Management Console 1656
Bonnes pratiques relatives aux quotas de service Amazon ECS et aux limites de limitation des API 1657
Elastic Load Balancing 1659
Interfaces réseau Elastic 1660
AWS Cloud Map 1662
Référence d'API Amazon ECS 1664
Historique du document 1665
Glossaire AWS 1709
..... mdccx

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

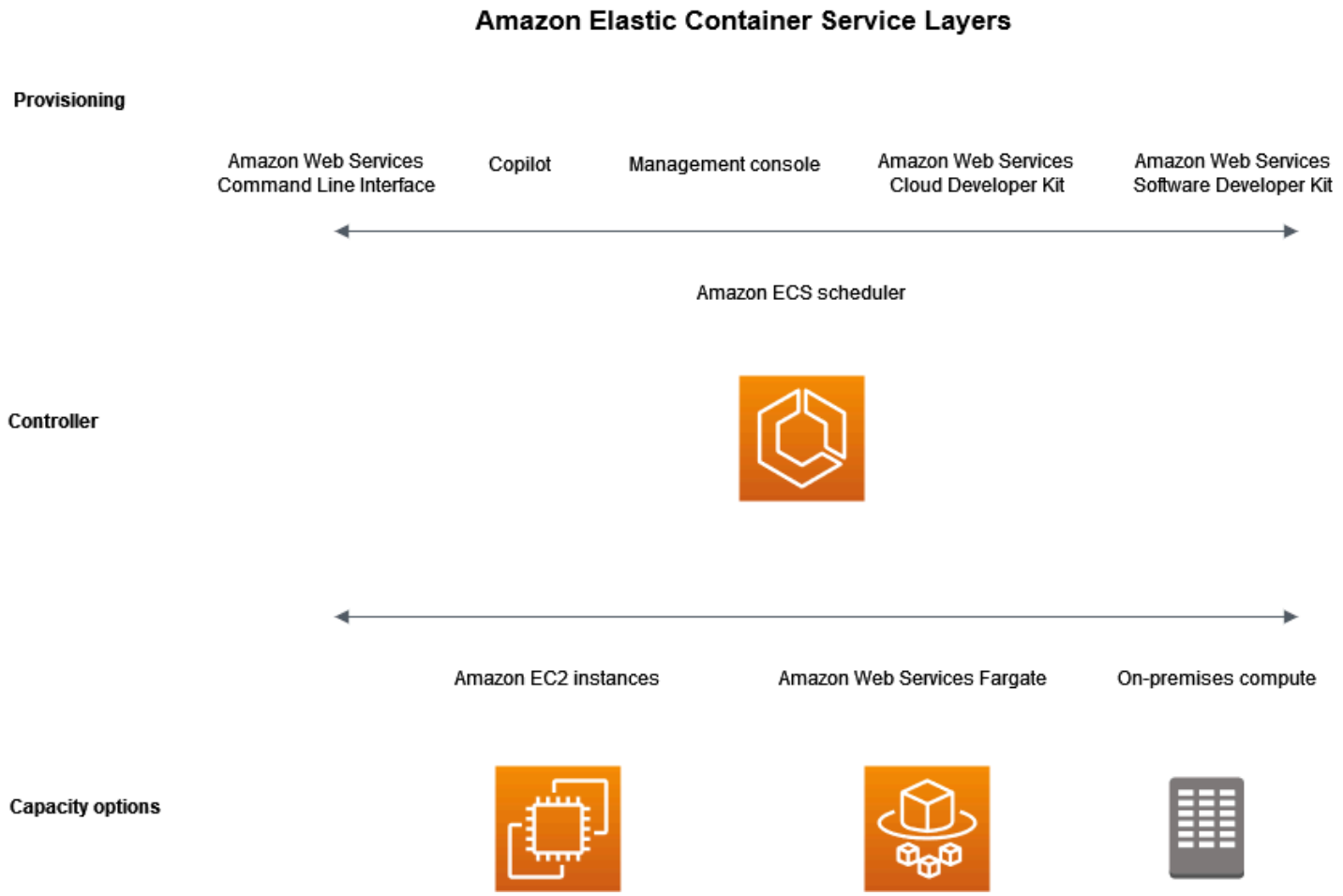
Amazon Elastic Container Service (Amazon ECS) est un service d'orchestration de conteneurs entièrement géré qui vous permet de déployer, de gérer et de dimensionner aisément des applications conteneurisées. En tant que service entièrement géré, Amazon ECS intègre les meilleures pratiques opérationnelles et de AWS configuration. Il est intégré à la fois à des outils AWS et à des outils tiers, tels qu'Amazon Elastic Container Registry et 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

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.



Capacité Amazon ECS

La capacité Amazon ECS est l'infrastructure sur laquelle vos conteneurs s'exécutent. Voici un aperçu des options de capacité :

- Instances Amazon EC2 dans le cloud AWS

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

La capacité peut être localisée dans l'une des AWS ressources suivantes :

- Zones de disponibilité
- Local Zones
- Zones Wavelength
- Régions AWS
- AWS Outposts

Contrôleur Amazon ECS

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

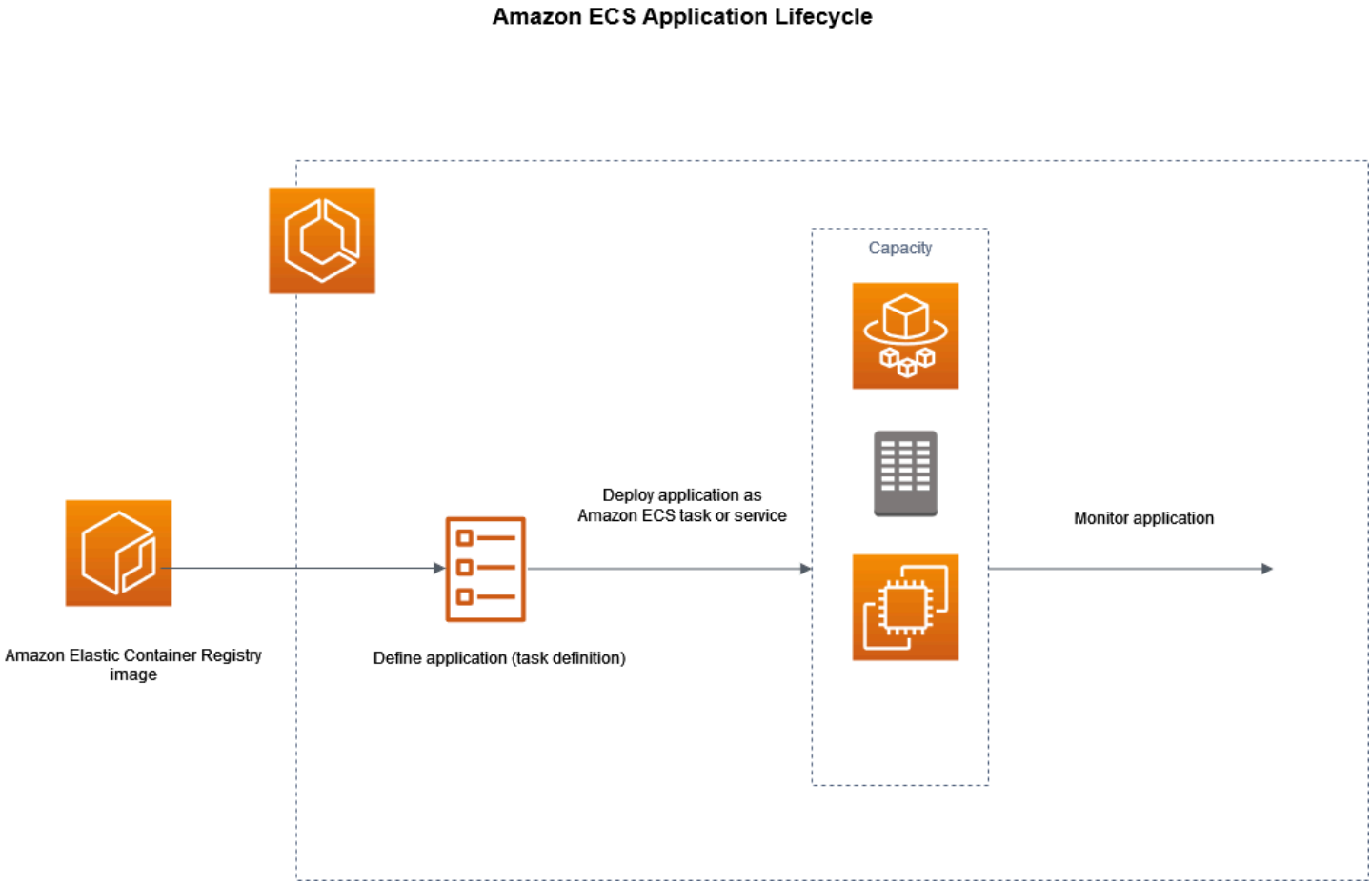
Provisionnement Amazon ECS

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 plus d'informations, consultez AWS Command Line Interface.
- AWS SDK — Fournit des API spécifiques au langage 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 plus d'informations, consultez Kits SDK AWS.
- Copilot : fournit un outil open source permettant aux développeurs de créer, de publier et d'exploiter des applications conteneurisées prêtes à la production sur Amazon ECS. Pour plus d'informations, consultez Copilot sur le GitHub site Web.
- 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.

Cycle de vie des applications

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 fichier Dockerfile est un fichier en texte brut qui spécifie tous les éléments qui sont inclus dans le 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

Informations connexes

Les ressources connexes suivantes peuvent s'avérer utiles lors de l'utilisation de ce service.

- AWS Fargate : vue d'ensemble des fonctionnalités de Fargate.
- Windows activé AWS : présentation de Windows sur les AWS charges de travail et les services.
- Linux d' AWS : portefeuille de systèmes d'exploitation modernes basés sur Linux d' AWS.

Didacticiels pour les développeurs

- AWS Blogues informatiques : informations sur les nouvelles fonctionnalités, informations détaillées sur les fonctionnalités, exemples de code et meilleures pratiques.

AWS re:Post

AWS re:Post— un service AWS géré de questions et réponses (Q & A) offrant des réponses participatives et révisées par des experts à vos questions techniques.

Tarification

- Tarification Amazon ECS : informations sur la tarification pour Amazon ECS.
- Tarification AWS Fargate : informations sur la tarification pour Fargate.

AWS Ressources générales

Les ressources générales suivantes peuvent vous aider dans votre travail avec AWS.

- Cours et ateliers — Liens vers des cours spécialisés et basés sur des rôles, ainsi que des ateliers à votre rythme pour vous aider à perfectionner vos AWS compétences et à acquérir une expérience pratique.
- AWS Centre pour développeurs : découvrez les didacticiels, téléchargez des outils et découvrez les événements AWS destinés aux développeurs.
- AWS Outils de développement : liens vers des outils de développement, des SDK, des boîtes à outils IDE et des outils de ligne de commande pour le développement et la gestion AWS d'applications.
- Centre de ressources pour la mise en route : découvrez comment configurer votre application Compte AWS, rejoindre la AWS communauté et lancer votre première application.
- Tutoriels pratiques — Suivez les step-by-step didacticiels pour lancer votre première application sur AWS.
- AWS Livres blancs : liens vers une liste complète de livres AWS blancs techniques, traitant de sujets tels que l'architecture, la sécurité et l'économie, rédigés par des architectes de AWS solutions ou d'autres experts techniques.
- AWS Support Centre — Le centre de création et de gestion de vos AWS Support dossiers. Comprend également des liens vers d'autres ressources utiles, telles que des forums, des FAQ techniques, l'état de santé du service et AWS Trusted Advisor.
- AWS Support— La principale page Web contenant des informations sur AWS Support un one-on-one canal d'assistance à réponse rapide pour vous aider à créer et à exécuter des applications dans le cloud.
- Contactez-nous : point de contact central pour toute question relative à la facturation AWS , à votre compte, aux événements, à des abus ou à d'autres problèmes.

- AWS Conditions du site — Informations détaillées sur nos droits d'auteur et notre marque commerciale ; votre compte, votre licence et l'accès au site ; et d'autres sujets.

Mise en route avec Amazon ECS

Les guides suivants fournissent une introduction aux outils disponibles pour accéder à Amazon ECS, ainsi que des procédures d'introduction étape par étape 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
- Commencer à utiliser les conteneurs Linux sur AWS Fargate
- Commencer à utiliser Windows sur Amazon EC2

Configurer l'utilisation d'Amazon ECS

Si vous êtes déjà inscrit à Amazon Web Services (AWS) et que vous avez utilisé Amazon Elastic Compute Cloud (Amazon EC2), vous pouvez presque 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.

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 https://portal.aws.amazon.com/billing/signup.
2. Suivez les instructions en ligne.

   Dans le cadre de la procédure d’inscription, vous recevrez un appel téléphonique 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 root 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 administratif, et à uniquement utiliser l'utilisateur root pour effectuer tâches nécessitant un accès utilisateur root.

AWS vous envoie un e-mail de confirmation une fois le processus d'inscription terminé. Vous pouvez afficher l'activité en cours de votre compte et gérer votre compte à tout moment en accédant à https://aws.amazon.com/ et en cliquant sur Mon compte.

Création d'un utilisateur administratif

Après vous être inscrit à un Compte AWS, sécurisez Utilisateur racine d'un compte AWS AWS IAM Identity Center, activez et créez un utilisateur administratif 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 root, consultez Connexion en tant qu'utilisateur root dans le Guide de l'utilisateur Connexion à AWS .
2. Activez l'authentification multifactorielle (MFA) pour votre utilisateur root.

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

Création d'un utilisateur 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 administratif.

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 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.

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, veuillez consulter Plateformes prises en charge sur la console Amazon EC2 dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux. Sinon, vous pouvez créer dans votre compte un VPC autre qu'un VPC par défaut en suivant les étapes ci-après.

> **Important**
>
> Si votre compte prend en charge Amazon EC2 Classic dans une région, vous n'avez pas de VPC par défaut dans cette région.

Pour plus d'informations sur la création d'un VPC, veuillez consulter Créer un VPC uniquement dans le Guide de l'utilisateur Amazon VPC et utilisez le tableau suivant pour déterminer quelles options sélectionner.

Option	Valeur
Ressources à créer	VPC uniquement
Nom	Vous pouvez, si vous le souhaitez, nommer votre VPC.
Bloc d'adresse CIDR IPv4	Entrée manuelle CIDR IPv4 La taille du bloc d'adresse CIDR doit être comprise entre /16 et /28.
Bloc d'adresse CIDR IPv6	Pas de bloc d'adresse CIDR IPv6
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 de plus amples 'informations, veuillez consulter Régions et zones de disponibilité dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

> **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 Création d'un groupe de sécurité dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux et utilisez le tableau suivant pour déterminer quelles options 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	VPC par défaut (marqué par « (par défaut) »). **Note** Si votre compte prend en charge Amazon EC2 Classic, sélectionnez le VPC créé lors de l'étape précédente.

Pour obtenir des informations sur les règles de trafic sortant à ajouter pour vos cas d'utilisation, veuillez consulter Règles de groupe de sécurité pour différents cas d'utilisation dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

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 le mode de création d'un groupe de sécurité, consultez la section Ajout de règles à votre groupe de sécurité dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

Option	Valeur	
Règle HTTP	Type : HTTP Source : n'importe où (`0.0.0.0/0`) Cette option ajoute automatiquement le bloc d'adresse CIDR IPv4 0.0.0.0/0 en tant que 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	

Option	Valeur	
	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 IP ou une plage d'adresses spécifiques.	

Option	Valeur
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.

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

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

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 le faire à l'aide de la console Amazon EC2. Si vous prévoyez de lancer des instances dans plusieurs régions, vous devez créer une paire de clés dans chaque région. Pour de plus amples informations sur les régions, veuillez consulter Régions et zones de disponibilité dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

Création d’une paire de clés

- Utilisez la console Amazon EC2 pour créer une paire de clés. Pour plus d'informations sur la création d'une paire de clés, consultez Create a key pair (Création d'une paire de clés) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

Pour plus d'informations sur la façon de se connecter à votre instance, consultez Connect to your Linux instance dans le guide de l'utilisateur Amazon EC2 pour les instances Linux.

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 bureau 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, consultez la section Installation de l'interface de ligne de AWS commande dans le guide de AWS Command Line Interface l'utilisateur.

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. Docker fournit une démonstration du déploiement de conteneurs sur Amazon ECS. Pour plus d'informations, consultez Docker Compose CLI - Amazon ECS.

L'objectif des étapes décrites ici est de vous guider lors de la création de votre première image Docker et de sa transmission 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 présentation de Docker.

Prérequis

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

- Veillez à avoir terminé les étapes de configuration d'Amazon ECR. Pour en savoir plus, consultez Configuration pour Amazon ECR dans le Guide de l'utilisateur 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 les étapes d'installation de Docker pour Amazon Linux 2, consultez Installation de Docker sur AL2023. 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 en savoir plus, consultez Installation d' AWS Command Line Interface dans le Guide de l'utilisateur AWS Command Line Interface .

Si vous n'avez pas besoin d'un environnement de développement local et que vous préférez utiliser une instance Amazon EC2 pour utiliser Docker, nous proposons la procédure suivante pour lancer une instance Amazon EC2 à l'aide d'Amazon Linux 2 et installer Docker Engine et l'interface de ligne de commande Docker.

Installation de Docker sur AL2023

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 déjà Amazon EC2, vous pouvez lancer une instance Amazon Linux 2023 et installer Docker pour démarrer.

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

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

1. Lancez une instance avec la dernière AMI Amazon Linux 2023. Pour en savoir plus, consultez Lancement d'une instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.
2. Connectez-vous à votre instance. Pour en savoir plus, consultez Connexion à votre instance Linux dans le guide de l'utilisateur Amazon EC2 pour les instances Linux.
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 Docker pour lancer des conteneurs sur des instances de conteneur dans vos clusters. Dans cette section, vous créez une image Docker pour une simple application web et vous la testez sur votre système local ou l'instance Amazon EC2, puis transférez l'image vers le registre de conteneur 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

   # Install dependencies
   RUN yum update -y && \
    yum install -y httpd

   # Install apache and 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 fichier Dockerfile utilise l'image publique Amazon Linux 2 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'EXPOSEinstruction signifie que le port 80 du conteneur est celui qui écoute, et l'CMDinstruction 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.

```
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. Pour en savoir plus sur docker run, accédez à Docker run reference.

```
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 instance EC2, il s'agit de la valeur de DNS public du serveur, c'est-à-dire la même adresse que celle que vous utilisez pour vous connecter à l'instance avec le protocole 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/.
   - Si vous l'utilisez docker-machine sur un ordinateur Windows ou Mac, recherchez l'adresse IP de la VirtualBox machine virtuelle hébergeant Docker à l'aide de la docker-machine ip commande, en remplaçant *`machine-name par le nom`* de la machine docker que vous utilisez.

     ```
     docker-machine ip machine-name
     ```

   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 AWS Docker géré. Vous pouvez utiliser la CLI Docker pour transmettre, extraire et gérer les images dans vos référentiels Amazon ECR. Pour consulter des détails sur le produit Amazon ECR, des études de cas de clients et des FAQ, rendez-vous sur les pages détaillées du produit 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, consultez Installation d' AWS Command Line Interface dans le Guide de l'utilisateur AWS Command Line Interface .

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 plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Une fois que vous avez créé et transféré votre image de conteneur vers Amazon ECR, vous devez prendre en compte les étapes suivantes.

- the section called “Utilisation de la console avec des conteneurs Linux sur AWS Fargate”
- the section called “Utilisation de conteneurs Windows sur AWS Fargate”
- Création d'un cluster avec une tâche Fargate Linux à l'aide du AWS CLI

Commencer à utiliser les conteneurs Linux sur AWS 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. Amazon ECS sur AWS 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 AWS utilisateur 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 plus d'informations, consultez 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 plus d'informations, consultez 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, veuillez consulter Ajout de règles à votre groupe de sécurité dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux (langue française non garantie).
>
> - 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 plus d'informations, consultez the section called "Exemples de cluster".

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 instances Amazon EC2, les équilibreurs de charge Elastic Load Balancing et les groupes Auto Scaling, doivent être nettoyées manuellement dans la console Amazon EC2 ou en supprimant la pile qui les AWS CloudFormation 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 delete ***cluster-name***, puis choisissez Delete (Supprimer). La suppression du cluster nettoie les ressources associées qui ont été créées avec le cluster, y compris les groupes Auto Scaling, les VPC ou les équilibreurs de charge.

Commencer à utiliser Windows sur Amazon EC2

Commencez à utiliser Amazon ECS en utilisant le type de lancement EC2 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 lancement EC2.

Prérequis

Avant de commencer, suivez les étapes ci-dessous Configurer l'utilisation d'Amazon ECS et assurez-vous que votre AWS utilisateur 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 plus d’informations, consultez 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 plus d'informations, consultez 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, veuillez consulter Ajout de règles à votre groupe de sécurité dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux (langue française non garantie).

- 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 avec une instance Amazon EC2 enregistrée dans celui-ci, 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 plus d'informations, consultez the section called “Exemples de cluster”.

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 instances Amazon EC2 à votre cluster, développez Infrastructure, puis sélectionnez Instances Amazon EC2. 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.
      - Pour EC2 instance type (Type d'instance EC2), choisissez le type d'instance pour vos charges de travail. Pour plus d'informations sur les différents types d'instances et leurs cas d'utilisation, veuillez consulter Instances Amazon EC2.

        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. Les instances Amazon EC2 entraînent des coûts tant qu'elles existent dans vos ressources. AWS Pour plus d’informations, consultez Tarification Amazon EC2.

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 Launch type (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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.

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 instances Amazon EC2, les équilibreurs de charge Elastic Load Balancing et les groupes Auto Scaling, doivent être nettoyées manuellement dans la console Amazon EC2 ou en supprimant la pile qui les AWS CloudFormation 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 delete ***`cluster-name`***, puis choisissez Delete (Supprimer). La suppression du cluster nettoie les ressources associées qui ont été créées avec le cluster, y compris les groupes Auto Scaling, les VPC ou les équilibreurs de charge.

Présentation des outils pour développeur Amazon ECS

Que vous fassiez partie d'une grande entreprise ou d'une start-up, Amazon ECS propose une variété d'outils qui peuvent vous aider à mettre vos conteneurs en service rapidement, quel que soit votre niveau d'expertise. Vous pouvez utiliser Amazon ECS de l'une des façons suivantes.

- Découvrez, développez, gérez et visualisez vos applications et services de conteneur à l'aide de la AWS Management Console.
- Effectuez des actions spécifiques sur les ressources Amazon ECS avec des déploiements automatisés via la programmation ou les scripts à l'aide d'AWS Command Line Interface, des kits Kits SDK AWS ou de l'API ECS.
- Définissez et gérez toutes les ressources AWS de votre environnement avec un déploiement automatisé à l'aide d'AWS CloudFormation.
- Utilisez le flux de travail complet des CLI AWS Copilot end-to-end développeurs pour créer, publier et exploiter des applications de conteneur conformes aux AWS meilleures pratiques en matière d'infrastructure.
- À l'aide de votre langage de programmation préféré, définissez l'infrastructure ou l'architecture en tant que code avec AWS CDK.
- Conteneurisez les applications hébergées sur site ou sur des instances Amazon EC2 (ou les deux) à l'aide de la portabilité intégrée et de l'écosystème d'outils d'AWS App2Container pour les conteneurs.
- Déployez une application sur Amazon ECS ou testez des conteneurs locaux avec des conteneurs s'exécutant dans Amazon ECS en utilisant le format de fichier Docker Compose, à l'aide de l'CLI Amazon ECS.
- Lancez des conteneurs à partir de l'Intégration de Docker Desktop avec Amazon ECS à l'aide d'Amazon ECS dans Docker Desktop.

AWS Management Console

La AWS Management Console est une interface basée sur un navigateur qui permet 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 Mise en route avec Amazon ECS.

Au début, de nombreux clients préfèrent utiliser la console, car elle fournit un retour visuel instantané sur la réussite des actions effectuées. Les clients AWS familiarisés avec la AWS Management Console peuvent facilement gérer les ressources associées telles que les équilibreurs de charge et les instances Amazon EC2.

Commencez avec la AWS Management Console.

AWS Command Line Interface

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

AWS fournit deux jeux d'outils de ligne de commande : l'AWS Command Line Interface (AWS CLI) et les AWS Tools for Windows PowerShell. Pour plus d'informations, consultez le Guide de l'utilisateur AWS Command Line Interface et le Guide de l'utilisateur AWS Tools for Windows PowerShell.

La AWS CLI convient aux clients qui préfèrent et sont habitués à l'écriture de scripts et à l'interfaçage avec un outil de ligne de commande et qui savent exactement quelles actions ils veulent effectuer sur leurs ressources Amazon ECS. La AWS CLI est également utile pour les clients qui souhaitent se familiariser avec les API Amazon ECS. Les clients peuvent utiliser la AWS CLI pour effectuer un certain nombre d'opérations sur les ressources Amazon ECS, notamment les opérations Create, Read, Update et Delete, directement à partir de l'interface de ligne de commande.

Utilisez la AWS CLI si vous êtes familiarisé ou souhaitez vous familiariser avec les API Amazon ECS et les commandes de la CLI correspondantes et que vous souhaitez écrire des scripts automatisés et effectuer des actions spécifiques sur les ressources Amazon ECS.

AWS CloudFormation

AWS CloudFormation et Terraform pour Amazon ECS sont des outils efficaces pour définir votre Infrastructure as Code. Vous pouvez facilement savoir à tout moment quelle version de votre modèle ou de votre pile AWS CloudFormation est en cours d'exécution et revenir à une version précédente si nécessaire. Vous pouvez effectuer des déploiements d'infrastructure et d'applications de manière automatisée également. C'est cette flexibilité et à cette automatisation qui font de

AWS CloudFormation et Terraform deux formats populaires pour le déploiement d'applications sur Amazon ECS à partir de pipelines de livraison continue.

Pour plus d'informations sur AWS CloudFormation, consultez Création de ressources Amazon ECS avec AWS CloudFormation.

Utilisez AWS CloudFormation ou Terraform si vous souhaitez automatiser les déploiements d'infrastructure et les applications sur Amazon ECS et définir et gérer explicitement toutes les ressources AWS de votre environnement.

CLI AWS Copilot

La CLI (interface de ligne de commande) AWS Copilot est un outil complet qui permet aux clients de déployer et d'exploiter des applications empaquetées dans des conteneurs et des environnements sur Amazon ECS directement à partir de leur code source. Avec AWS Copilot, vous pouvez effectuer ces opérations sans avoir besoin de comprendre AWS et les éléments Amazon ECS tels que les équilibreurs Application Load Balancers, les sous-réseaux publics, les tâches, les services et les clusters. AWS Copilot crée des ressources AWS en votre nom à partir de modèles de service d'opinion, tels qu'un service web ou un service backend à charge équilibrée, fournissant un environnement de production immédiat pour les applications conteneurisées. Vous pouvez déployer via un pipeline AWS CodePipeline sur plusieurs environnements, comptes ou régions, qui peuvent tous être gérés au sein de la CLI. Avec AWS Copilot, vous pouvez également effectuer des tâches d'opérateur, telles que l'affichage des journaux et de l'état de votre service. AWS Copilot est un all-in-one outil qui vous aide à gérer plus facilement vos ressources cloud afin que vous puissiez vous concentrer sur le développement et la gestion de vos applications.

Pour plus d’informations, consultez Utilisation de l'interface de ligne de commande AWS Copilot.

Utilisez le flux de travail de end-to-end développement complet de AWS Copilot pour créer, publier et exploiter des applications de conteneur conformes aux AWS meilleures pratiques en matière d'infrastructure.

AWS CDK

L'AWS Cloud Development Kit (AWS CDK) est 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. AWS CDK alloue vos ressources de manière sûre et reproductible à l'aide d'AWS CloudFormation. Grâce à CDK, les clients

peuvent générer leur environnement avec moins de lignes de code en utilisant le même langage que celui utilisé pour créer leur application. Amazon ECS fournit un module dans CDK nommé `ecs-patterns`, qui crée des architectures communes. Un modèle disponible est `ApplicationLoadBalancedFargateService()`. Ce modèle crée un cluster, une définition de tâche et d'autres ressources pour exécuter un service Amazon ECS service à charge équilibrée sur AWS Fargate.

Pour plus d'informations, consultez Mise en route avec Amazon ECS à l'aide du AWS CDK.

Utilisez AWS CDK si vous souhaitez définir l'infrastructure ou l'architecture en tant que code dans votre langage de programmation préféré. Par exemple, vous pouvez utiliser le même langage que celui de vos applications.

AWS App2Container

Parfois, les clients d'entreprise peuvent déjà avoir des applications hébergées sur site ou sur des instances EC2 (ou les deux). Ils s'intéressent à la portabilité et à l'écosystème d'outils des conteneurs, en particulier sur Amazon ECS, et doivent d'abord conteneuriser. AWS C'est exactement ce qu'App2Container vous permet de faire. App2Container (A2C) est un outil de ligne de commande qui modernise les applications .NET et Java en applications conteneurisées. A2C analyse et crée un inventaire de toutes les applications exécutées sur des machines virtuelles, sur site ou dans le cloud. Une fois que vous avez sélectionné l'application que vous voulez conteneuriser, A2C empaquète l'artefact de l'application et les dépendances identifiées dans des images de conteneur. Il configure ensuite les ports réseau et génère la tâche Amazon ECS. Enfin, il crée un CloudFormation modèle que vous pouvez déployer ou modifier si nécessaire.

Pour de plus amples informations, veuillez consulter Getting started with AWS App2Container.

Utilisez App2Container si vous avez des applications hébergées sur site ou sur des instances Amazon EC2 ou les deux.

CLI Amazon ECS

La CLI Amazon ECS vous permet d'exécuter vos applications sur Amazon ECS et AWS Fargate à l'aide du format de fichier Docker Compose. Vous pouvez rapidement allouer des ressources, importer et télécharger des images à l'aide d'Amazon ECR et surveiller les applications en cours d'exécution sur Amazon ECS ou AWS Fargate. Vous pouvez également tester des conteneurs s'exécutant localement avec des conteneurs dans le cloud au sein de la CLI.

Pour plus d’informations, consultez Utilisation de l'interface de ligne de commande Amazon ECS.

Utilisez la CLI ECS si vous disposez d'une application Compose et souhaitez la déployer sur Amazon ECS, ou tester des conteneurs locaux avec des conteneurs s'exécutant dans Amazon ECS dans le cloud.

Intégration de Docker Desktop avec Amazon ECS

AWS et Docker ont collaboré pour créer une expérience de développement simplifiée que vous pouvez utiliser pour déployer et gérer des conteneurs sur Amazon ECS directement à l'aide d'outils Docker. Vous pouvez désormais créer et tester vos conteneurs localement à l'aide de Docker Desktop et Docker Compose, puis les déployer vers Amazon ECS sur Fargate. Pour démarrer l'intégration d'Amazon ECS et de Docker, téléchargez Docker Desktop et inscrivez-vous éventuellement pour recevoir un ID Docker. Pour de plus amples informations, veuillez consulter Docker Desktop et Docker ID signup.

Les personnes qui débutent avec les conteneurs commencent généralement avec des outils Docker tels que la CLI Docker et Docker Compose. Ainsi, le plugin de la CLI Docker Compose pour Amazon ECS est l'étape naturelle suivante dans l'exécution de conteneurs sur AWS après les avoir testés localement. Docker fournit une démonstration du déploiement de conteneurs sur Amazon ECS. Pour plus d'informations, consultez Docker Compose CLI - Amazon ECS.

Vous pouvez tirer parti des autres fonctions Amazon ECS, telles que la découverte de service, la répartition de charge et d'autres ressources AWS à utiliser avec leurs applications avec Docker Desktop.

Vous pouvez également télécharger le plug-in Docker Compose CLI pour Amazon ECS directement depuis GitHub. Pour plus d'informations, consultez le plug-in Docker Compose CLI pour Amazon ECS sur GitHub.

Kits SDK AWS

Vous pouvez également utiliser les kits SDK AWS pour gérer les ressources et les opérations Amazon ECS à partir de différents langages de programmation. Les kits SDK fournissent des modules pour vous aider à prendre en charge les tâches, y compris les tâches de la liste suivante :

- Signature cryptographique des requêtes de service
- Nouvelles tentatives de requête

- Gestion des réponses d'erreur

Pour en savoir plus sur les kits SDK disponibles, consultez Outils pour Amazon Web Services.

Récapitulatif

Choisissez parmi nos nombreuses options celle qui vous convient le mieux. Prenez en compte les options suivantes.

- Si vous avez une orientation visuelle, vous pouvez créer et exploiter visuellement des conteneurs à l'aide de la AWS Management Console.
- Si vous préférez les CLI, envisagez d'utiliser AWS Copilot ou la AWS CLI. Sinon, si vous préférez l'écosystème Docker, vous pouvez tirer parti des fonctionnalités d'ECS à partir de la CLI Docker pour déployer vers AWS. Une fois ces ressources déployées, vous pouvez continuer à les gérer via la CLI ou visuellement via la console.
- Si vous êtes un développeur, vous pouvez utiliser AWS CDK pour définir votre infrastructure dans le même langage que votre application. Vous pouvez utiliser le CDK et AWS Copilot pour exporter vers des CloudFormation modèles dans lesquels vous pouvez modifier les paramètres granulaires, ajouter d'autres AWS ressources et automatiser les déploiements par le biais de scripts ou d'un pipeline CI/CD tel que. AWS CodePipeline

La AWS CLI, les kits SDK et les API ECS sont des outils utiles pour automatiser les actions sur les ressources ECS, ce qui les rend idéales pour le déploiement. Pour déployer des applications à l'aide d'AWS CloudFormation, vous pouvez utiliser une variété de langages de programmation ou un simple fichier texte pour modéliser et allouer toutes les ressources nécessaires à vos applications. Vous pouvez ensuite déployer votre application sur plusieurs régions et comptes de manière automatisée et sécurisée. Par exemple, vous pouvez définir votre cluster ECS, vos services, vos définitions de tâches ou vos fournisseurs de capacité sous forme de code dans un fichier et les déployer via les AWS CLI CloudFormation commandes.

Pour effectuer des tâches d'opérations, vous pouvez afficher et gérer les ressources par programme à l'aide de la AWS CLI, du kit SDK ou de l'API ECS. Les commandes comme `describe-tasks` ou `list-services` affichent les dernières métadonnées ou une liste de toutes les ressources. Comme pour les déploiements, les clients peuvent écrire une automatisation qui inclut des commandes telles que `update-service` pour fournir des actions correctives lors de la détection d'une ressource qui s'est arrêtée de façon inattendue. Vous pouvez également exploiter vos services à l'aide d'AWS

Copilot. Les commandes telles que `copilot svc logs` ou `copilot app show` fournissent des détails sur chacun de vos microservices ou sur votre application dans son ensemble.

Les clients peuvent utiliser n'importe lequel des outils disponibles mentionnés dans ce document et les utiliser dans diverses combinaisons. Les outils ECS offrent plusieurs moyens de passer d'un outil à l'autre en fonction de l'évolution de vos besoins. Par exemple, vous pouvez opter pour un contrôle plus détaillé des ressources ou à plus d'automatisation. ECS offre également une large gamme d'outils pour un large éventail de besoins et de niveaux d'expertise.

Utilisation de l'interface de ligne de commande AWS Copilot

Les commandes de l'interface de ligne de commande (CLI) d'AWS Copilot simplifient la création, la publication et l'exploitation des applications conteneurisées prêtes pour la production sur Amazon ECS à partir d'un environnement de développement local. La CLI AWS Copilot s'aligne sur les flux de développement qui prennent en charge les bonnes pratiques d'applications modernes : de l'utilisation de l'Infrastructure as Code à la création d'un pipeline CI/CD alloué pour le compte d'un utilisateur. Utilisez la CLI AWS Copilot dans le cadre du cycle de tests et de développement quotidien, à la place de la AWS Management Console.

AWS Copilot prend actuellement en charge les systèmes Linux, macOS et Windows. Pour plus d'informations sur la version la plus récente da la CLI AWS Copilot, consultez Releases.

> **Note**
>
> Le code source de la AWS CLI Copilot est disponible sur. GitHub La documentation la plus récente de la CLI est disponible sur le site web d'AWS Copilot. 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, GitHuboù vous pouvez signaler des problèmes, faire part de vos commentaires et signaler des bogues.

Installation de la CLI AWS Copilot

La CLI AWS Copilot peut être installée sur des systèmes Linux ou macOS en utilisant Homebrew ou en téléchargeant manuellement le fichier binaire. Suivez les étapes suivantes avec la méthode d'installation que vous préférez.

Installation de la CLI AWS Copilot en utilisant Homebrew

La commande suivante est utilisée pour installer la CLI AWS 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
```

Installation manuelle de la CLI AWS Copilot

Comme alternative à Homebrew, vous pouvez installer manuellement la CLI AWS Copilot sur votre système macOS ou Linux. Utilisez la commande suivante pour que votre système d'exploitation télécharge le fichier binaire, appliquez-lui les autorisations d'exécution, puis vérifiez qu'il fonctionne en répertoriant le menu d'aide.

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 fichiers exécutables de la CLI AWS Copilot sont signés de manière chiffrée à l'aide de signatures PGP. Les signatures PGP peuvent être utilisées pour vérifier la validité du fichier 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
```

Vous pouvez fermer l'éditeur de texte.

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 CLI AWS 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 d'AWS Copilot au chemin d'accès.

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

Étapes suivantes

Après l'installation, découvrez comment déployer une application Amazon ECS à l'aide d'AWS Copilot. Pour plus d'informations, consultez Mise en route avec Amazon ECS à l'aide d'AWS Copilot.

Mise en route avec Amazon ECS à l'aide d'AWS Copilot

Démarrez avec Amazon ECS à l'aide d'AWS Copilot en déployant une application Amazon ECS.

Prérequis

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

- Configurez un compte AWS. Pour plus d'informations, consultez Configurer l'utilisation d'Amazon ECS.
- Installez la CLI AWS Copilot. Les versions prennent actuellement en charge les systèmes Linux et macOS. Pour plus d'informations, consultez Installation de la CLI AWS Copilot.
- 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 CLI AWS Copilot utilisera pour gérer votre application et vos services.
- Installez et exécutez Docker. Pour plus d'informations, veuillez consulter Get started with Docker.

Déployer votre application à l'aide d'une commande

Assurez-vous que l'outil de ligne de commande AWS est installé et que vous avez déjà exécuté `aws configure` avant de démarrer.

Déployez l'application à l'aide de la commande suivante.

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

Déployer votre application étape par étape

Étape 1 : Configurer vos informations d'identification

Exécutez `aws configure` pour configurer un profil par défaut que la CLI AWS Copilot utilise pour gérer votre application et vos services.

```
aws configure
```

Étape 2 : Cloner l'application de démonstration

Clonez une application Flask simple et Dockerfile.

```
git clone https://github.com/aws-samples/amazon-ecs-cli-sample-app.git demo-app
```

Étape 3 : Configurer votre application

1. À partir du répertoire demo-app, exécutez la commande `init`.

   Pour les utilisateurs de Windows, exécutez la commande `init` depuis le dossier qui contient le fichier téléchargé `copilot.exe`.

   ```
   copilot init
   ```

AWS Copilot vous guide à travers la configuration de votre première application et premier service avec une série d'invites de terminal, en commençant par l'étape suivante. Si vous avez déjà utilisé AWS Copilot pour déployer des applications, vous êtes invité à en choisir une dans une liste de noms d'applications.

2. Nommez votre application.

```
What would you like to name your application? [? for help]
```

Enter ***demo***.

Étape 4 : Configurer un service ECS dans votre application de « démonstration »

1. Vous êtes invité à choisir un type de service. Vous construisez une application Flask simple qui sert une petite API.

```
Which service type best represents your service's architecture? [Use arrows to
 move, type to filter, ? for more help]
  > Load Balanced Web Service
    Backend Service
    Scheduled Job
```

Choose ***Load Balanced Web Service*** .

2. Indiquez un nom pour votre service.

```
What do you want to name this Load Balanced Web Service? [? for help]
```

Enter ***api*** pour le nom de votre service.

3. Sélectionnez un Dockerfile.

```
Which Dockerfile would you like to use for api? [Use arrows to move, type to
 filter, ? for more help]
  > ./Dockerfile
    Use an existing image instead
```

Choose ***Dockerfile***.

Pour les utilisateurs de Windows, saisissez le chemin d'accès au Dockerfile dans la `demo-app folder` (*`... \ demo-app \ Dockerfile`* \.).

4. Définissez un port.

```
Which port do you want customer traffic sent to? [? for help] (80)
```

Enter ***80*** ou acceptez la valeur par défaut.

5. Un journal indiquant les ressources de l'application en cours de création apparaît.

```
Creating the infrastructure to manage services under application demo.
```

6. Une fois les ressources de l'application créées, déployez un environnement de test.

```
Would you like to deploy a test environment? [? for help] (y/N)
```

Enter ***y***.

```
Proposing infrastructure changes for the test environment.
```

7. Un journal affichant l'état du déploiement de votre application s'affiche.

```
Note: It's best to run this command in the root of your Git repository.
Welcome to the Copilot CLI! We're going to walk you through some questions
to help you get set up with an application on ECS. An application is a collection
 of
containerized services that operate together.

Use existing application: No
Application name: demo
Workload type: Load Balanced Web Service
Service name: api
Dockerfile: ./Dockerfile
no EXPOSE statements in Dockerfile ./Dockerfile
Port: 80
Ok great, we'll set up a Load Balanced Web Service named api in application demo
 listening on port 80.

# Created the infrastructure to manage services under application demo.

# Wrote the manifest for service api at copilot/api/manifest.yml
```

```
Your manifest contains configurations like your container size and port (:80).

# Created ECR repositories for service api.

All right, you're all set for local development.
Deploy: Yes

# Created the infrastructure for the test environment.
- Virtual private cloud on 2 availability zones to hold your services
 [Complete]
- Virtual private cloud on 2 availability zones to hold your services
 [Complete]
  - Internet gateway to connect the network to the internet
 [Complete]
  - Public subnets for internet facing services
 [Complete]
  - Private subnets for services that can't be reached from the internet
 [Complete]
  - Routing tables for services to talk with each other
 [Complete]
- ECS Cluster to hold your services
 [Complete]
# Linked account aws_account_id and region region to application demo.

# Created environment test in region region under application demo.

Environment test is already on the latest version v1.0.0, skip upgrade.
[+] Building 0.8s (7/7) FINISHED
 => [internal] load .dockerignore
                                0.1s
 => => transferring context: 2B
                                0.0s
 => [internal] load build definition from Dockerfile
                                0.0s
 => => transferring dockerfile: 37B
                                0.0s
 => [internal] load metadata for docker.io/library/nginx:latest
                                0.7s
 => [internal] load build context
                                0.0s
 => => transferring context: 32B
                                0.0s
```

```
 => [1/2] FROM docker.io/library/
nginx@sha256:aeade65e99e5d5e7ce162833636f692354c227ff438556e5f3ed0335b7cc2f1b
 0.0s
 => CACHED [2/2] COPY index.html /usr/share/nginx/html
                                 0.0s
 => exporting to image
                                 0.0s
 => => exporting layers
                                 0.0s
 => => writing image
 sha256:3ee02fd4c0f67d7bd808ed7fc73263880649834cbb05d5ca62380f539f4884c4
           0.0s
 => => naming to aws_account_id.dkr.ecr.region.amazonaws.com/demo/api:cee7709
                                 0.0s
WARNING! Your password will be stored unencrypted in /home/user/.docker/
config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
The push refers to repository
 [aws_account_id.dkr.ecr.region.amazonaws.com/demo/api]
592a5c0c47f1: Pushed
6c7de695ede3: Pushed
2f4accd375d9: Pushed
ffc9b21953f4: Pushed
cee7709: digest: sha_digest

# Deployed api, you can access it at http://demo-
Publi-1OQ8VMS2VC2WG-561733989.region.elb.amazonaws.com.
```

Étape 5 : Vérifier que votre application est en cours d'exécution

Consultez l'état de votre application à l'aide des commandes suivantes.

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
```

Répertoriez les commandes et les options disponibles.

```
copilot --help
```

```
copilot init --help
```

Étape 6. Apprendre à créer un pipeline CI/CD

Les instructions sont disponibles dans Atelier ECS, qui explique en détail la manière d'automatiser complètement un pipeline CI/CD et un flux de travail git à l'aide d'AWS Copilot.

Étape 7 : Nettoyer

Exécutez la commande suivante pour supprimer et nettoyer toutes les ressources.

```
copilot app delete
```

Mise en route avec Amazon ECS à l'aide du AWS CDK

Le AWS Cloud Development Kit (AWS CDK) est un cadre IAC (Infrastructure-as-Code) qui vous permet de définir l'infrastructure de cloud AWS en utilisant un 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 à un modèle AWS CloudFormation et déployez vos ressources dans votre Compte AWS. Suivez les étapes de cette rubrique pour déployer un serveur web conteneurisé avec Amazon Elastic Container Service (Amazon ECS) et le AWS CDK sur Fargate.

L'AWS Construct Library, incluse avec le CDK, fournit des modules que vous pouvez utiliser pour modéliser les ressources fournies par les Services AWS. 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 module `aws-ecs-patterns` inclut également des constructions qui utilisent un Network Load Balancer et s'exécutent sur Amazon EC2.

Avant de commencer cette tâche, configurez votre environnement de développement AWS CDK et installez leAWS CDK en exécutant la commande suivante. Pour de plus amples informations sur la configuration de votre environnement de développement AWS CDK, consultez Mise en route avec le AWS CDK - Prérequis.

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

> **Note**
>
> Ces instructions considèrent que vous utilisez v2 AWS CDK.

Rubriques

- Étape 1 : Configurer votre projet AWS CDK
- Étape 2 : Utiliser AWS CDK pour définir un serveur web conteneurisé sur Fargate

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

Étape 1 : Configurer votre projet AWS CDK

Créez un répertoire pour votre nouvelle appli AWS CDK 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 les dépendances de base de AWS CDK.

```
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
```

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

v2 AWS CDK inclut des constructions stables pour tous les 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
```

Étape 2 : Utiliser AWS CDK pour définir un serveur web conteneurisé sur Fargate

Utilisez l'image du conteneur `amazon-ecs-sample`provenant de DockerHub. Cette image contient une application web PHP fonctionnant sous Amazon Linux 2.

Dans le project AWS CDK que vous avez créé, modifiez le fichier contenant la définition de 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 application AWS CDK peut contenir plusieurs piles, tandis que 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('amazon/amazon-ecs-sample'),
      },
      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
                });
        }
    }
}
```

Le court extrait précédent inclut les éléments suivants :

- Le nom logique du service : `MyWebServer`.
- L'image du conteneur obtenue à partir de DockerHub :`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.

Le 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
- Instances Amazon VPC et 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. Le AWS CDK exécute l'application et en synthétise un modèle AWS CloudFormation et 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 comande `cdk deploy` dans le répertoire principal de votre application. Il vous est demandé d'approuver les stratégies IAM que le AWS CDK a généré.

Le déploiement prend plusieurs minutes pendant lesquelles le AWS CDK crée plusieurs ressources. 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 de l'infrastructure AWS à l'aide du AWS CDK, veuillez consulter le Guide du développeur AWS CDK.

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

TypeScript

- Travailler avec le AWS CDK in TypeScript

JavaScript

- Travailler avec le AWS CDK in JavaScript

Python

- Utilisation du AWS CDK dans Python

Java

- Utilisation du AWS CDK dans Java

C#

- Utilisation du AWS CDK dans C#

Pour plus d'informations sur les modules AWS Construct Library utilisés dans cette rubrique, veuillez consulter les présentations des Références d'API AWS CDK ci-dessous.

- aws-ecs
- aws-ecs-patterns

Création de ressources Amazon ECS avec AWS CloudFormation

Amazon ECS est intégré à AWS CloudFormation, un service que vous pouvez utiliser pour modéliser et à configurer les ressources AWS avec des modèles que vous définissez. Ainsi, vous pouvez consacrer moins de temps à la création et à la gestion de vos ressources et de votre infrastructure. A l'aide de AWS CloudFormation, vous pouvez créer un modèle qui décrit tous les ressources AWS que vous souhaitez, telles que des clusters Amazon ECS spécifiques. Ensuite, AWS CloudFormation s'occupe pour vous de la mise en service et de la configuration de ces ressources.

Lorsque vous utilisez AWS CloudFormation, vous pouvez réutiliser votre modèle pour configurer vos ressources Amazon ECS de manière cohérente et reproductible. Vous décrivez vos ressources une seule fois, puis allouez à nouveau les mêmes ressources sur plusieurs Comptes AWS et Régions AWS.

Amazon ECS et modèles AWS CloudFormation

Pour allouer et configurer des ressources pour Amazon ECS et les services associés, assurez-vous que vous connaissez bien les modèles AWS CloudFormation. Les modèles AWS CloudFormation sont des fichiers texte au format JSON ou YAML qui décrivent les ressources que vous souhaitez allouer dans vos piles AWS CloudFormation. Si le format JSON ou YAML (ou les deux) ne vous est pas familier, vous pouvez utiliser le AWS CloudFormation Designer pour commencer à utiliser les modèles AWS CloudFormation. Pour plus d’informations, consultez Qu’est-ce que AWS CloudFormation Designer ? dans le AWS CloudFormationGuide de l’utilisateur.

Amazon ECS prend en charge la création de clusters, de définitions de tâches, de services et de jeux de tâches dans AWS CloudFormation. Les exemples suivants montrent comment créer des ressources avec ces modèles à l'aide du AWS CLI. Vous pouvez également créer ces ressources à l'aide de la console AWS CloudFormation. Pour de plus amples informations sur la création de ressources à l'aide de la console AWS CloudFormation, consultez le AWS CloudFormationGuide de l'utilisateur.

Exemple de modèles

Création de ressources Amazon ECS à l'aide de piles distinctes

Les exemples suivants montrent comment créer des ressources Amazon ECS en utilisant des piles distinctes pour chaque ressource.

Définitions de tâche Amazon ECS

Vous pouvez utiliser le modèle suivant pour créer une tâche Fargate Linux.

JSON

```
  {
    "AWSTemplateFormatVersion": "2010-09-09",
    "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": "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::aws_account_id:role/
ecsTaskExecutionRole",
                "Family": "task-definition-cfn",
                "Memory": 512,
                "NetworkMode": "awsvpc",
                "RequiresCompatibilities": [
                    "FARGATE"
                ],
                "RuntimePlatform": {
                    "OperatingSystemFamily": "LINUX"
                }
            }

        }
        }
    }
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
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: '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::aws_account_id:role/ecsTaskExecutionRole'
    Family: task-definition-cfn
    Memory: 512
    NetworkMode: awsvpc
    RequiresCompatibilities:
      - FARGATE
    RuntimePlatform:
      OperatingSystemFamily: LINUX
```

Clusters Amazon ECS

Vous pouvez utiliser le modèle suivant pour créer un cluster vide.

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
```

Création de plusieurs ressources Amazon ECS dans une seule pile

Vous pouvez utiliser l'exemple de modèle suivant pour créer plusieurs ressources Amazon ECS dans une seule pile. Le modèle crée un cluster Amazon ECS nommé `CFNCluster`. Le cluster contient une définition de tâche Linux Fargate qui configure un serveur Web. Le modèle crée également un service nommé `cfn-service` qui lance et conserve la tâche définie par la définition de tâche. Avant d'utiliser ce modèle, assurez-vous que tous les ID de sous-réseau et de groupe de sécurité dans la `NetworkConfiguration` du service appartiennent au même VPC et que le groupe de sécurité dispose des règles nécessaires. Pour plus d'informations sur les règles des groupes de sécurité, veuillez consulter Règles des groupes de sécurité dans le Guide de l'utilisateur Amazon VPC.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": "CFNCluster"
            }
        },
        "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": "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::aws_account_id::role/
ecsTaskExecutionRole",
                "Family": "task-definition-cfn",
                "Memory": 512,
                "NetworkMode": "awsvpc",
                "RequiresCompatibilities": [
                    "FARGATE"
                ],
                "RuntimePlatform": {
                    "OperatingSystemFamily": "LINUX"
                }
            }
        },
        "ECSService": {
            "Type": "AWS::ECS::Service",
```

```
            "Properties": {
                "ServiceName": "cfn-service",
                "Cluster": {
                    "Ref": "ECSCluster"
                },
                "DesiredCount": 1,
                "LaunchType": "FARGATE",
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "AssignPublicIp": "ENABLED",
                        "SecurityGroups": [
                            "sg-abcdef01234567890"
                        ],
                        "Subnets": [
                            "subnet-abcdef01234567890"
                        ]
                    }
                },
                "TaskDefinition": {
                    "Ref": "ECSTaskDefinition"
                }
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  ECSCluster:
    Type: 'AWS::ECS::Cluster'
    Properties:
      ClusterName: CFNCluster
  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: '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::aws_account_id:role/ecsTaskExecutionRole'
    Family: task-definition-cfn
    Memory: 512
    NetworkMode: awsvpc
    RequiresCompatibilities:
      - FARGATE
    RuntimePlatform:
      OperatingSystemFamily: LINUX
ECSService:
  Type: 'AWS::ECS::Service'
  Properties:
    ServiceName: cfn-service
    Cluster: !Ref ECSCluster
    DesiredCount: 1
    LaunchType: FARGATE
    NetworkConfiguration:
      AwsvpcConfiguration:
        AssignPublicIp: ENABLED
        SecurityGroups:
          - sg-abcdef01234567890
        Subnets:
          - subnet-abcdef01234567890
```

```
      TaskDefinition: !Ref ECSTaskDefinition
```

Utilisation de AWS CLI pour créer des ressources à partir de modèles

La commande suivante permet de créer une pile nommée `ecs-stack` à l'aide d'un fichier de corps de modèle nommé `ecs-template-body.json`. Assurez-vous que le fichier de corps du modèle est au format JSON ou YAML. L'emplacement du fichier est spécifié dans le paramètre `--template-body`. Dans ce cas, le fichier de corps du modèle se trouve dans le répertoire actuelnt.

```
aws cloudformation create-stack \
      --stack-name ecs-stack \
      --template-body file://ecs-template-body.json
```

Pour vous assurer que les ressources sont créées correctement, consultez la console Amazon ECS ou utilisez les commandes suivantes :

- La commande suivante permet d'afficher toutes les définitions de tâches.

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

- La commande suivante permet d'afficher tous les clusters.

  ```
  aws ecs list-clusters
  ```

- La commande suivante permet d'afficher tous les services définis dans le cluster *`CFNCluster`*. Remplacez *`CFNCluster`* par le nom du cluster dans lequel vous souhaitez créer le service.

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

En savoir plus sur AWS CloudFormation

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

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

Utilisation de l'interface de ligne de commande Amazon ECS

Amazon ECS a publié AWS Copilot, un outil d'interface de ligne de commande (CLI) qui simplifie la création, la publication et l'exploitation des applications conteneurisées prêtes pour la productio n sur Amazon ECS à partir d'un environnement de développement local. Pour plus d'informations, consultez Utilisation de l'interface de ligne de commande AWS Copilot.

L'interface de ligne de commande (CLI) d'Amazon Elastic Container Service (Amazon ECS) fournit des commandes de haut niveau pour simplifier la création, la mise à jour et la surveillance des clusters et des tâches depuis un environnement de développement local. La CLI Amazon ECS prend en charge les fichiers Docker Compose, une spécification open source populaire qui permet de définir et d'exécuter des applications à conteneurs multiples. Utilisez la CLI ECS dans le cadre du cycle de tests et de développement quotidien, à la place de la AWS Management Console.

La dernière version de la CLI Amazon ECS ne prend en charge que les versions majeures de la syntaxe de fichier Docker Compose versions 1, 2 et 3. La version spécifiée dans le fichier Compose doit être la chaîne `"1"`, `"1.0"`, `"2"`, `"2.0"`, `"3"` ou `"3.0"`. Les versions mineures de Docker Compose ne sont pas prises en charge.

Le code source de la CLI Amazon ECS est disponible sur GitHub. Cet outil n'est plus activement développé.

Installation de la CLI Amazon ECS

Amazon ECS a publié AWS Copilot, un outil d'interface de ligne de commande (CLI) qui simplifie la création, la publication et l'exploitation des applications conteneurisées prêtes pour la productio n sur Amazon ECS à partir d'un environnement de développement local. Pour plus d'informations, consultez Utilisation de l'interface de ligne de commande AWS Copilot.

Les étapes suivantes démontrent comment installer la CLI Amazon ECS sur votre système macOS, Linux ou Windows.

Pour installer la CLI Amazon ECS

1. Téléchargez le fichier binaire de la CLI Amazon ECS.

macOS

```
sudo curl -Lo /usr/local/bin/ecs-cli https://amazon-ecs-cli.s3.amazonaws.com/
ecs-cli-darwin-amd64-latest
```

Linux

```
sudo curl -Lo /usr/local/bin/ecs-cli https://amazon-ecs-cli.s3.amazonaws.com/
ecs-cli-linux-amd64-latest
```

Windows

Ouvrez Windows PowerShell et entrez les commandes suivantes.

> **Note**
>
> Si vous rencontrez des problèmes d'autorisation, assurez-vous que vous disposez d'un accès administrateur sous Windows et que vous vous présentez PowerShell en tant qu'administrateur.

```
New-Item -Path 'C:\Program Files\Amazon\ECSCLI' -ItemType Directory
Invoke-WebRequest -OutFile 'C:\Program Files\Amazon\ECSCLI\ecs-cli.exe' https://
amazon-ecs-cli.s3.amazonaws.com/ecs-cli-windows-amd64-latest.exe
```

2. Vérifiez la CLI Amazon ECS à l'aide de signatures PGP. Les fichiers exécutables de la CLI Amazon ECS sont signés de manière chiffrée à l'aide de signatures PGP. Les signatures PGP peuvent être utilisées pour vérifier la validité du fichier exécutable de la CLI Amazon ECS. Utilisez les étapes suivantes pour vérifier les signatures à l'aide de l'outil GnuPG.

   a. 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.

b. 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>"
```

c. 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`

d. 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
```

Vous pouvez fermer l'éditeur de texte.

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

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

f. Téléchargez les signatures de la CLI Amazon ECS. 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

```
curl -Lo ecs-cli.asc https://amazon-ecs-cli.s3.amazonaws.com/ecs-cli-darwin-amd64-latest.asc
```

Linux

```
curl -Lo ecs-cli.asc https://amazon-ecs-cli.s3.amazonaws.com/ecs-cli-linux-amd64-latest.asc
```

Windows

```
Invoke-WebRequest -OutFile ecs-cli.asc https://amazon-ecs-cli.s3.amazonaws.com/ecs-cli-windows-amd64-latest.exe.asc
```

g. Vérifiez la signature.

macOS and Linux

```
gpg --verify ecs-cli.asc /usr/local/bin/ecs-cli
```

Windows

```
gpg --verify ecs-cli.asc 'C:\Program Files\Amazon\ECSCLI\ecs-cli.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.

3. Appliquez les autorisations d'exécution au fichier binaire.

   macOS and Linux

   ```
   sudo chmod +x /usr/local/bin/ecs-cli
   ```

   Windows

   Modifiez les variables d'environnement et ajoutez `C:\Program Files\Amazon\ECSCLI` dans le champ de la variable `PATH`, en les séparant des entrées existantes avec un point-virgule. Par exemple :

   ```
   setx path "%path%;C:\Program Files\Amazon\ECSCLI"
   ```

   Redémarrez PowerShell pour que les modifications entrent en vigueur.

   > **Note**
   >
   > Une fois la `PATH` variable définie, l'interface de ligne de commande Amazon ECS peut être utilisée depuis Windows PowerShell ou depuis l'invite de commande.

4. Vérifiez si la CLI fonctionne correctement.

   ```
   ecs-cli --version
   ```

   Passez à Configuration de la CLI Amazon ECS.

> **Important**
>
> Pour pouvoir utiliser la CLI Amazon ECS, vous devez d'abord la configurer avec vos informations d'identification AWS, une région AWS et un nom de cluster Amazon ECS. Pour plus d'informations, consultez Configuration de la CLI Amazon ECS.

Configuration de la CLI Amazon ECS

Amazon ECS a publié AWS Copilot, un outil d'interface de ligne de commande (CLI) qui simplifie la création, la publication et l'exploitation des applications conteneurisées prêtes pour la productio n sur Amazon ECS à partir d'un environnement de développement local. Pour plus d'informations, consultez Utilisation de l'interface de ligne de commande AWS Copilot.

Pour que vous puissiez l'utiliser, la CLI Amazon ECS exige quelques informations de configuration de base, telles que vos informations d'identification AWS, la région AWS dans laquelle créer le cluster et le nom du cluster Amazon ECS à utiliser. Les informations de configuration sont stockées dans le répertoire `~/.ecs` sur Les systèmes Mac OS et Linux et dans `C:\Users\<username>\AppData\local\ecs` sur les systèmes Windows.

Pour configurer la CLI Amazon ECS

1. Configurez un profil de CLI avec la commande suivante, en remplaçant *`profile_name`* par le nom de profil de votre choix, et les variables d'environnement *`$AWS_ACCESS_KEY_ID`* et *`$AWS_SECRET_ACCESS_KEY`* par vos informations d'identification AWS.

   ```
   ecs-cli configure profile --profile-name profile_name --access-
   key $AWS_ACCESS_KEY_ID --secret-key $AWS_SECRET_ACCESS_KEY
   ```

2. Terminez la configuration avec la commande suivante, en remplaçant *`launch_type`* par le type de lancement de tâche que vous souhaitez utiliser par défaut, *`region_name`* par la Région AWS de votre choix, *`cluster_name`* par le nom d'un cluster Amazon ECS existant ou d'un nouveau cluster à utiliser et *`configuration_name`* par le nom que vous souhaitez donner à cette configuration.

```
ecs-cli configure --cluster cluster_name --default-launch-type launch_type --
region region_name --config-name configuration_name
```

Utilisation de profils

La CLI Amazon ECS prend en charge la configuration de plusieurs ensembles d'informations d'identification AWS en tant que profiles nommés à l'aide de la commande ecs-cli configure profile. Un profil par défaut peut être défini avec la commande ecs-cli configure profile default. Ces profils peuvent ensuite être référencés lorsque vous exécutez des commandes CLI Amazon ECS qui nécessitent des informations d'identification à l'aide de l'indicateur `--ecs-profile`. Sinon, le profil par défaut est utilisé.

Utilisation de configurations de clusters

Une configuration de cluster est un ensemble de champs qui décrit un cluster Amazon ECS, y compris le nom du cluster et la région. Une configuration de cluster par défaut peut être définie avec la commande ecs-cli configure default. La CLI Amazon ECS prend en charge la configuration de plusieurs configurations de cluster nommés à l'aide de l'option `--config-name`.

Comprendre l'ordre de priorité

Il existe plusieurs méthodes pour transmettre les informations d'identification et la région dans une commande de la CLI Amazon ECS. Voici l'ordre de priorité de chacun de ces éléments.

L'ordre de priorité des informations d'identification est le suivant :

1. Indicateurs de profil de CLI Amazon ECS :
   a. Profil Amazon ECS (`--ecs-profile`)
   b. Profil AWS (`--aws-profile`)
2. Variables d'environnement :
   a. `ECS_PROFILE`
   b. `AWS_PROFILE`
   c. `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY` et `AWS_SESSION_TOKEN`
3. Configuration ECS : tente de récupérer les informations d'identification à partir du profil ECS par défaut.

4. Profil AWS par défaut : tente d'utiliser les informations d'identification (`aws_access_key_id`, `aws_secret_access_key`) ou `assume_role` (`role_arn`, `source_profile`) à partir du nom de profil AWS.
   a. Variable d'environnement `AWS_DEFAULT_PROFILE` (`default` par défaut).
5. Rôle d'instance EC2

L'ordre de priorité des informations de la région est le suivant :

1. Indicateurs de CLI Amazon ECS :
   a. Indicateur de région (`--region`)
   b. Indicateur de configuration de cluster (`--cluster-config`)
2. Configuration ECS : tente de récupérer la région à partir du profil ECS par défaut.
3. Variables d'environnement : tente de récupérer la région à partir des variables d'environnement suivantes :
   a. `AWS_REGION`
   b. `AWS_DEFAULT_REGION`
4. Profil AWS : tente d'utiliser la région à partir du profil AWS par défaut :
   a. variable d'environnement `AWS_PROFILE`
   b. variable d'environnement `AWS_DEFAULT_PROFILE` (par défaut `default`)

Amazon ECS sur AWS Fargate

AWS Fargate est une technologie que vous pouvez utiliser avec Amazon ECS pour exécuter des conteneurs sans avoir à gérer des serveurs ou des clusters d'instances Amazon EC2. 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 plus d'informations, consultez Types de lancement.

Fargate propose des versions de plateforme pour les éditions Amazon Linux 2 et 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 pas à pas de Fargate

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

- Commencer à utiliser les conteneurs Linux sur AWS Fargate
- Utilisation de conteneurs Windows sur AWS Fargate

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

- Création d'un cluster avec une tâche Fargate Linux à l'aide du AWS CLI
- Création d'un cluster avec une tâche Fargate Windows à l'aide du 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 plus d'informations, consultez AWS Fargate fournisseurs de capacité.

  Vous ne pouvez utiliser Fargate Spot pour les tâches Linux qui utilisent l'architecture X86.

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 plus d'informations, consultez 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 plus d'informations, consultez AWS FAQ sur la maintenance des tâches Fargate.

Pour plus d'informations, consultez Versions de plateforme Linux Fargate et Versions de plateformes Windows Fargate.

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 service sur AWS Fargate prennent en charge les équilibreurs de charge Application Load Balancer et Network Load Balancer. Les équilibreurs de charge Application Load Balancer permettent d'acheminer le trafic HTTP/HTTPS (ou de couche 7). Les équilibreurs de charge Network Load Balancer permettent d'acheminer le trafic TCP ou UDP (ou couche 4). Pour plus d'informations, consultez Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge.

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 réseau `awsvpc` sont associées à une interface réseau Elastic et non à une instance Amazon EC2. Pour plus d'informations, consultez Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge.

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 de plus amples informations sur les Service Quotas pour AWS Fargate, consultez AWS Fargate quotas de service.

Pour plus d'informations sur les métriques AWS Fargate d'utilisation, consultez les métriques AWS Fargate d'utilisation dans le guide de l'utilisateur d'Amazon Elastic Container Service pour AWS Fargate.

Versions de plateforme Linux Fargate

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 plus d'informations, consultez AWS FAQ sur la maintenance des tâches Fargate.

Considérations

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

- Lorsque vous spécifiez une version de plateforme, vous pouvez utiliser un numéro de version spécifique (par exemple, `1.4.0`) ou `LATEST`.

Lorsque la version la plus récente de la plateforme est sélectionnée, la version `1.4.0` est utilisée.

- 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 plateforme Linux `1.4.0`, vous pouvez mettre à jour votre service et spécifier une nouvelle version de plateforme. 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 Types de déploiement 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 s'exécutent toujours sur la dernière révision d'une version de plateforme, ce qui garantit que les tâches sont toujours lancées 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.

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 plus d’informations, consultez Transmission de données sensibles à un conteneur.

  - Spécifiez les variables d'environnement en bloc à l'aide du paramètre de définition de conteneur `environmentFiles`. Pour plus d'informations, consultez Utiliser les paramètres de définition des tâches pour transmettre des variables d'environnement à un conteneur.
  - Les tâches exécutées dans un VPC et un sous-réseau activés pour IPv6 se verront attribuer à la fois une adresse IPv4 et une adresse IPv6 privées. Pour de plus amples informations, veuillez consulter Mise en réseau des tâches Fargate dans le Guide de l'utilisateur Amazon Elastic Container Service pour AWS 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 plus d'informations, consultez Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge.
- 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 de la tâche Fargate et Utilisation de volumes de données dans des tâches.
- Ajout de la prise en charge de l'utilisation des volumes de système de fichiers Amazon EFS pour le stockage des tâches permanentes. Pour plus d'informations, consultez Volumes Amazon EFS.
- Le stockage des tâches éphémères est désormais d'au moins 20 Go pour chaque tâche. Pour plus d'informations, consultez Stockage éphémère de la tâche Fargate.
- 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 lancement Amazon EC2, consultez Fargate Task Networking. Pour plus d'informations sur la mise en réseau pour le type de lancement Fargate, consultez. Mise en réseau des tâches pour les tâches sur Fargate
- Les ENI des tâches incluent désormais la prise en charge des trames 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 mesures de performance réseau pour les tâches Fargate. Pour plus d'informations, consultez Surveillez les conteneurs Amazon ECS à l'aide de Container Insights.
- 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, voir > 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.
- Ajout de la prise en charge du paramètre `SYS_PTRACE` Linux dans les définitions de conteneur. Pour plus d'informations, consultez 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 Codes d'erreur pour les tâches arrêtées dans le Guide de l'utilisateur Amazon Elastic Container Service pour AWS Fargate.
- 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 plus d'informations, consultez Utilisation du routage personnalisé des journaux.

- 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 Maintenance de tâches dans le Guide de l'utilisateur Amazon Elastic Container Service pour AWS Fargate.
- 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 plus d'informations, consultez Transmission de données sensibles à un conteneur.
- 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 plus d'informations, consultez Transmission de données sensibles à un conteneur.
- 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 plus d'informations, consultez Stockage et journalisation.
- À compter du 9 juillet 2019, toutes les nouvelles tâches Fargate lancées CloudWatch seront prises en charge par Container Insights. Pour plus d'informations, consultez Surveillez les conteneurs Amazon ECS à l'aide de Container Insights.
- Depuis le 3 décembre 2019, le fournisseur de capacité Fargate Spot est pris en charge. Pour plus d'informations, consultez AWS Fargate fournisseurs de capacité.
- Basée sur Amazon Linux 2.

Migration vers la version 1.4.0 de la plateforme Linux

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`. Une bonne pratique consiste à vérifier que votre tâche fonctionne correctement sur la version `1.4.0` de la plateforme avant de la migrer.

- 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 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 Mise en réseau des tâches pour les tâches sur Fargate.
- Si vous utilisez des points de terminaison d'un VPC d'interface, tenez compte des éléments suivants.
  - Lorsque vous utilisez des images de conteneur hébergées avec Amazon ECR, les points de terminaison d'un VPC ECR com.amazonaws.region.ecr.dkr et com.amazonaws.region.ecr.api, ainsi que le point de terminaison de la passerelle Simple Storage Service (Amazon S3), sont requis. 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.
  - Lorsque vous utilisez une définition de tâche qui fait référence à des secrets gérés par Secrets Manager pour récupérer des données sensibles pour vos conteneurs, 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 .
  - Lorsque vous utilisez une définition de tâche qui fait référence à des paramètres Systems Manager Parameter Store pour récupérer des données sensibles pour vos conteneurs, vous devez créer les points de terminaison d'un VPC d'interface pour Systems Manager. Pour de plus amples informations, veuillez consulter Utilisation de Systems Manager avec des points de terminaison d'un VPC dans le Guide de l'utilisateur AWS Systems Manager .
  - Assurez-vous que le groupe de sécurité dans l'Elastic Network Interface (ENI) associé à votre tâche dispose des règles de groupe de sécurité créées pour autoriser le trafic entre la tâche et les points de terminaison de VPC que vous utilisez.

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.

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 plateforme Linux Fargate.

Changelog pour les versions obsolètes de Fargate Linux AWS

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 plus d'informations, consultez Authentification de registre privé pour les tâches.

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 plus d'informations, consultez 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 plus d'informations, consultez Surveillance de l'état.
- Ajout de la prise en charge de la découverte de service Amazon ECS service. Pour plus d'informations, consultez Découverte de service.

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.

Versions de plateformes Windows Fargate

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 plus d'informations, consultez AWS FAQ sur la maintenance des tâches Fargate.

Remarques relatives à la version de plateforme

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

- Lorsque vous spécifiez une version de plateforme, vous pouvez utiliser un numéro de version spécifique (par exemple, `1.0.0`) ou `LATEST`.

  Lorsque la version la plus récente de la plateforme est sélectionnée, la plateforme `1.0.0` est utilisée.
- Les nouvelles tâches s'exécutent toujours sur la dernière révision d'une version de plateforme, ce qui garantit que les tâches sont toujours lancées sur une infrastructure sécurisée et corrigée.
- 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.

- 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.

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 sur les conteneurs Windows sur Fargate

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 plus d'informations, consultez the section called "Utilisation du pilote du journal awslogs".

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

- Les comptes de service gérés de groupe (gMSA)
- Amazon FSx
- Jonction ENI
- Les intégrations du service App Mesh et du proxy pour les tâches
- L'intégration du routeur journal Firelens pour les tâches
- Les volumes EFS
- Les paramètres de définition de tâche suivants :
  - `maxSwap`
  - `swappiness`
- 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 plus d'informations, consultez Montages liés.

AWS FAQ sur la maintenance des tâches Fargate

Qu'est-ce que le maintien des tâches et la mise hors service de Fargate ?

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. 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. Dès 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 plus d'informations, consultez 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

Pour les tâches de service, vous n'avez aucune action à effectuer, 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. Par défaut, le pourcentage minimum de fonctionnement d'un service est de 100 %, de sorte qu'une nouvelle tâche est démarrée avant d'être arrêtée. Les tâches de service sont régulièrement remplacées de la même manière lorsque vous adaptez le service, que vous déployez des modifications de configuration ou que vous déployez des révisions de définitions de tâches. Afin de vous préparer au nouveau processus de retrait de tâche, nous vous recommandons de tester le comportement de votre application en simulant ce scénario. Pour ce faire, arrêtez une tâche dans votre service pour tester sa résilience.

Pour le retrait d'une tâche autonome, AWS arrête la tâche à la date de fin de la tâche ou après cette date. Nous ne lançons 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`.

Que contient l'avis de mise hors service des tâches ?

Les notifications de retrait des 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, les ID des tâches.
- Pour les tâches de service, l'ID du cluster sur lequel le service s'exécute et les ID 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": "cluster/service"
            },
            {
                "entityValue": "cluster/service"
            }
        ]
    }
}
```

- E-mails

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

Puis-je modifier le temps d'attente pour la retraite 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.

Pour plus d'informations, consultez AWS Fargate temps d'attente pour la retraite des tâches.

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 plus d'informations, consultez Puis-je modifier le temps d'attente pour la retraite des tâches ?. 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 AWS Support.

Puis-je contrôler le calendrier de remplacement d'une tâche ?

Pour les services qui utilisent le déploiement progressif, vous devez mettre à jour le service à l'`update-service`aide 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 déploiement bleu/vert, vous devez créer un nouveau déploiement dans. AWS CodeDeployPour plus d'informations sur la façon de créer le déploiement, voir create-deployment dans la AWS Command Line Interface référence.

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

Amazon ECS remplace progressivement les tâches concernées dans votre service au début de la période de mise hors service de Fargate. Lorsqu'Amazon ECS arrête une tâche, il utilise le

pourcentage de santé minimum du service et lance une nouvelle tâche afin de maintenir le nombre de tâches souhaité pour le service. Une nouvelle tâche est démarrée avant qu'une tâche ne soit arrêtée car le pourcentage de santé minimum par défaut est de 100. Les tâches de service sont régulièrement remplacées de la même manière lorsque vous adaptez le service, que vous déployez des modifications de configuration ou que vous déployez des révisions de définitions de tâches. Pour plus d'informations sur le pourcentage de santé minimal, consultezConfiguration de déploiement.

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.

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. Les ID de zone de disponibilité pris en charge sont indiqués 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
Afrique (Le Cap)	af-south-1
Asie-Pacifique (Hong Kong)	ap-east-1
Asie-Pacifique (Mumbai)	ap-south-1

Nom de la région	Région
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
Asie-Pacifique (Singapour)	ap-southeast-1
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 (`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

Nom de la région	Région
Europe (Stockholm)	eu-north-1
Amérique du Sud (São Paulo)	sa-east-1
Israël (Tel Aviv)	il-central-1
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. Les ID de zone de disponibilité pris en charge sont indiqués 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` uniquement)
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

Nom de la région	Région
Asie-Pacifique (Osaka)	ap-northeast-3
Asie-Pacifique (Séoul)	ap-northeast-2
Asie-Pacifique (Singapour)	ap-southeast-1
Asie-Pacifique (Sydney)	ap-southeast-2
Asie-Pacifique (Melbourne)	ap-southeast-4
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

Nom de la région	Région
Israël (Tel Aviv)	il-central-1
Moyen-Orient (EAU)	me-central-1
Moyen-Orient (Bahreïn)	me-south-1

Conception de votre solution pour Amazon ECS

Avant d'utiliser Amazon ECS, vous devez prendre des décisions concernant la capacité, le réseau, les paramètres du compte et la journalisation afin de pouvoir configurer correctement vos ressources Amazon ECS.

Capacité Amazon ECS

La capacité est l'infrastructure dans laquelle vos conteneurs fonctionnent. Les options sont les suivantes :

- Instances Amazon EC2
- Sans serveur ()AWS Fargate (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.

Réseaux

AWS les ressources sont créées dans des sous-réseaux. Lorsque vous utilisez des instances EC2, 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
- Local Zones
- 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.

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.
- `awsvpc`tronquage

  Pour certains types d'instances EC2, des interfaces réseau (ENI) supplémentaires 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 que`ecsCreateCluster`. Si des balises sont spécifiées dans l'action de création de ressources,

octroie AWS une autorisation supplémentaire à l'`ecs:TagResource`action 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 via IPv4, IPv6 ou les deux.

- Format Amazon Resource Name (ARN)

  Certaines fonctionnalités, telles que l'autorisation de balisage, nécessitent un nouveau format Amazon Resource Name (ARN).

Pour plus d’informations, consultez Accès aux fonctionnalités d'Amazon ECS via les paramètres du compte.

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.

Type de lancement Fargate

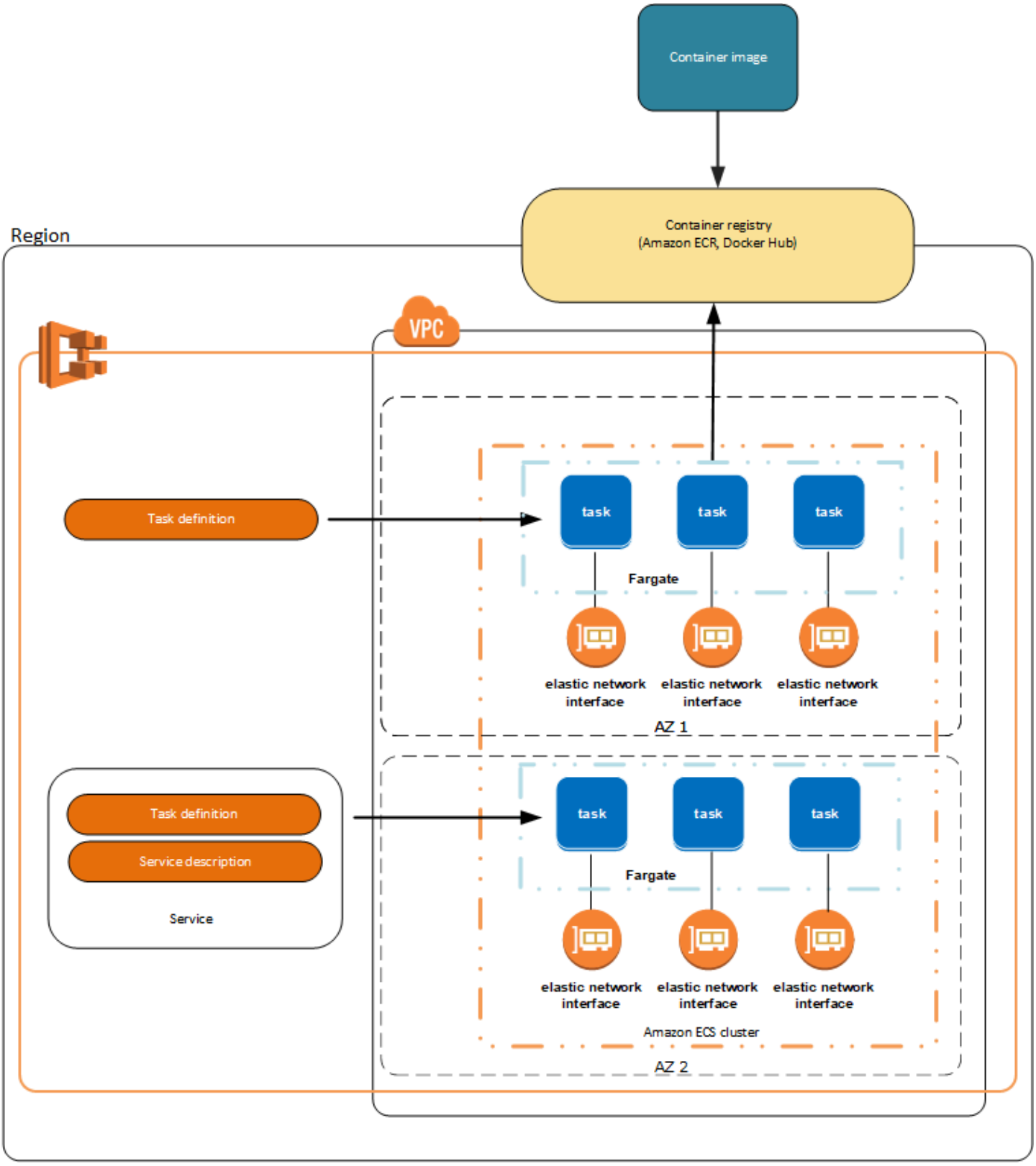
Vous pouvez utiliser le type de lancement Fargate pour exécuter vos applications conteneurisées sans avoir à provisionner ni à gérer l'infrastructure sous-jacente. AWS Fargate est le moyen sans serveur d'héberger vos charges de travail Amazon ECS.

Le type de lancement 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 [Amazon ECS sur AWS Fargate](#).

Type de lancement EC2

Le type de lancement EC2 convient aux charges de travail volumineuses dont le prix doit être optimisé.

Lorsque vous envisagez de modéliser les définitions de tâches et les services à l'aide du type de lancement EC2, nous vous recommandons de déterminer quels processus doivent s'exécuter ensemble et comment vous pouvez procéder pour mettre à l'échelle chaque composant.

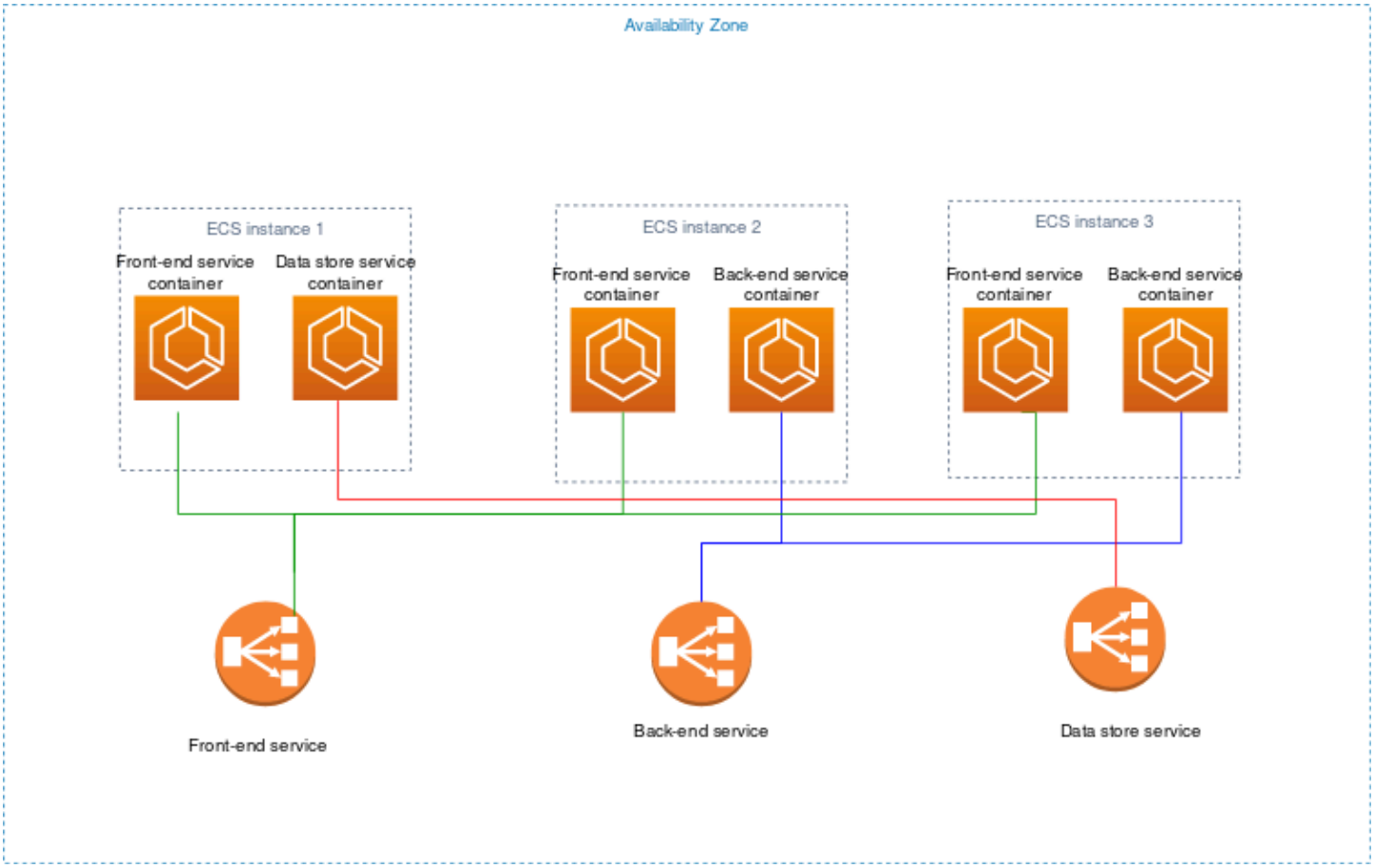
Par exemple, supposons qu'une application comporte les composants suivants :

- Un service frontal qui affiche des informations sur une page web
- Un service backend qui fournit les API pour le service frontal
- 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 plus d’informations, consultez Création d'un service à l'aide de la console. Dans vos services, vous pouvez associer des conteneurs avec des équilibreurs de charge Elastic Load Balancing. Pour plus d’informations, consultez Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge. 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 plus d’informations, consultez Mise à jour d'un service à l'aide de la console.



Type de lancement externe

Le type de lancement externe est utilisé pour exécuter vos applications conteneurisées sur votre serveur sur site ou machine virtuelle (VM) que vous enregistrez dans votre cluster Amazon ECS et que vous gérez à distance. Pour plus d'informations, consultez Instances externes (Amazon ECS Anywhere).

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 , consultez la section How should I think about when to use AWS Wavelength, AWS Local Zones ou AWS Outposts pour les applications nécessitant une faible latence ou un traitement local des données dans les AWS Wavelength FAQ.

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 des VPC partagés pour le type de lancement EC2 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 des VPC partagés 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 instances Amazon EC2, des serveurs de fichiers Amazon FSx et des Application Load Balancers à utiliser pour vos clusters et tâches Amazon ECS.

Pour de plus amples informations, veuillez consulter Local Zones dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

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 les instances Amazon EC2 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. Ensuite, vous pouvez lancer dans la zone vos instances Amazon EC2 à 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 API, 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.

Prérequis

Voici les prérequis à satisfaire pour utiliser Amazon ECS sur AWS Outposts :

- Vous devez avoir installé et configuré un AWS Outposts dans votre centre de données sur site.
- Vous devez disposer d'une connexion réseau fiable entre votre AWS région AWS Outposts et la sienne.
- Vous devez disposer d'une capacité suffisante de types d'instances dans votre AWS Outposts.
- Toutes les instances de conteneur Amazon ECS doivent disposer d'un agent de conteneur Amazon ECS version 1.33.0 ou ultérieure.

Limites

L'utilisation d'Amazon ECS sur AWS Outposts présente les limites suivantes :

- 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.

Considérations relatives à la connectivité réseau

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.

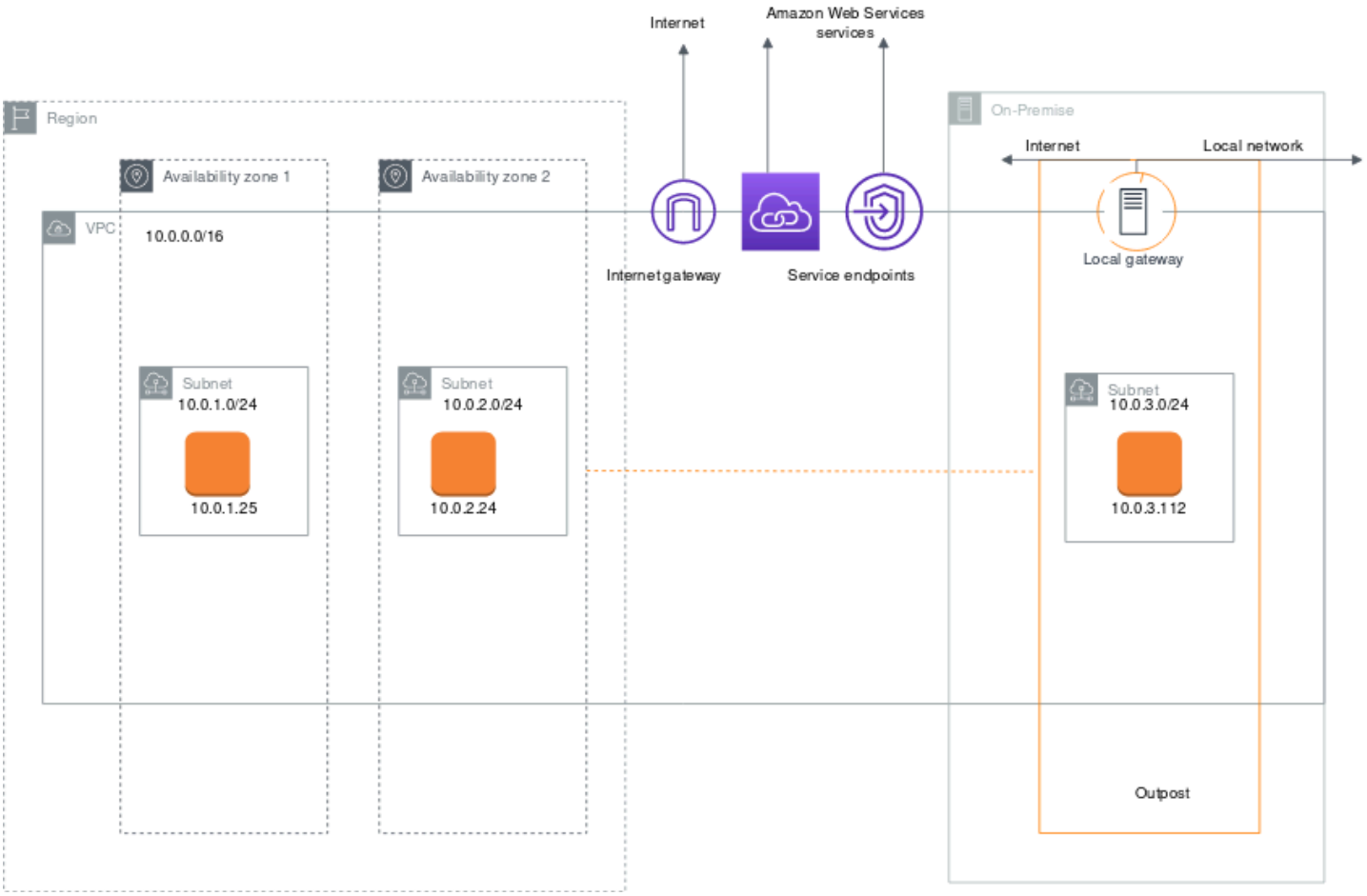
L'agent CloudWatch Logs ne sera pas en mesure de mettre à jour les journaux et les données des événements.

- Nous vous recommandons de fournir une connectivité fiable, à haute disponibilité et à faible latence entre votre AWS région AWS Outposts et la sienne.

Création d'un cluster Amazon ECS sur un AWS Outposts

La création d'un cluster Amazon ECS sur un AWS Outposts est similaire à la création d'un cluster Amazon ECS dans le AWS cloud. Lorsque vous créez un cluster Amazon ECS sur un AWS Outposts, vous devez spécifier un sous-réseau associé à votre AWS Outposts.

An AWS Outposts est une extension d'une AWS région, et vous pouvez étendre un Amazon VPC dans un compte pour couvrir plusieurs zones de disponibilité et toutes les zones associées. AWS Outposts Lorsque vous configurez votre AWS Outposts, vous y associez un sous-réseau pour étendre votre environnement VPC régional à votre installation sur site. Les instances présentes sur un AWS Outposts apparaissent dans le cadre de votre VPC régional, comme dans le cas d'une zone de disponibilité avec des sous-réseaux associés.



AWS CLI

Pour créer un cluster Amazon ECS sur et AWS Outposts avec le AWS CLI, spécifiez un groupe de sécurité et un sous-réseau à associer à votre AWS Outposts.

Pour créer 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-xxxxxxxxxxxxxxxxx \
    --availability-zone-id usw2-az1
```

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 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
```

5. Créez le cluster.
6. 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`.

7. 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 de plus amples informations sur le fichier des appels d'API, consultez Exécuter des commandes au lancement sur votre instance Linux dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

```
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.

8. Enregistrez votre définition de tâche. Utilisez la commande suivante et remplacez `ecs-task.json` par le nom de votre définition de tâche.

```
aws ecs register-task-definition --cli-input-json file://ecs-task.json
```

9. Exécutez la tâche ou créez le service.

   Run the task

```
aws ecs run-task --cluster mycluster --count 1 --task-definition outpost-app:1
```

   Create the service

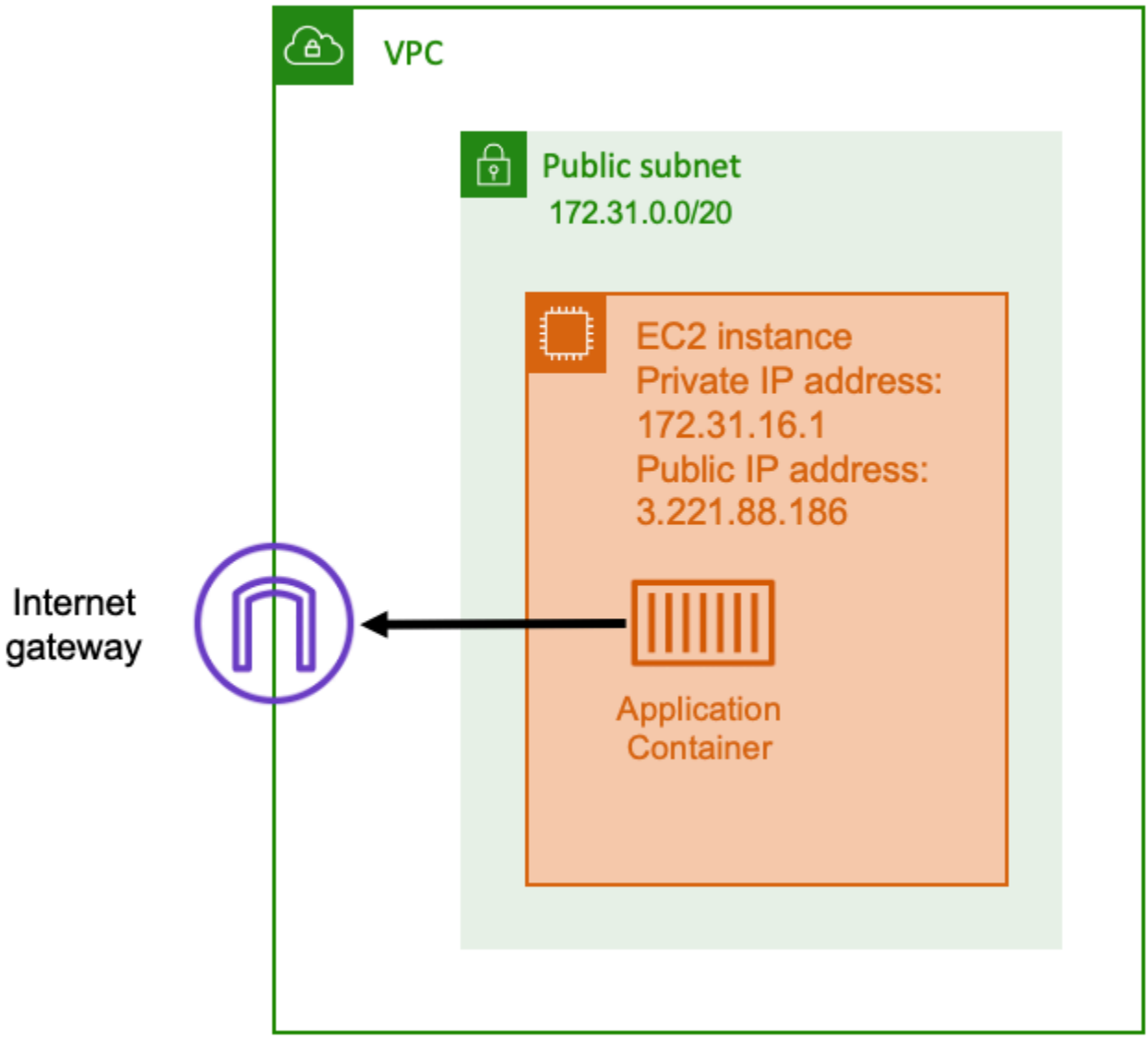
```
aws ecs create-service --cluster mycluster --service-name outpost-service \
    --task-definition outpost-app:1 --desired-count 1
```

Bonnes pratiques pour connecter 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.

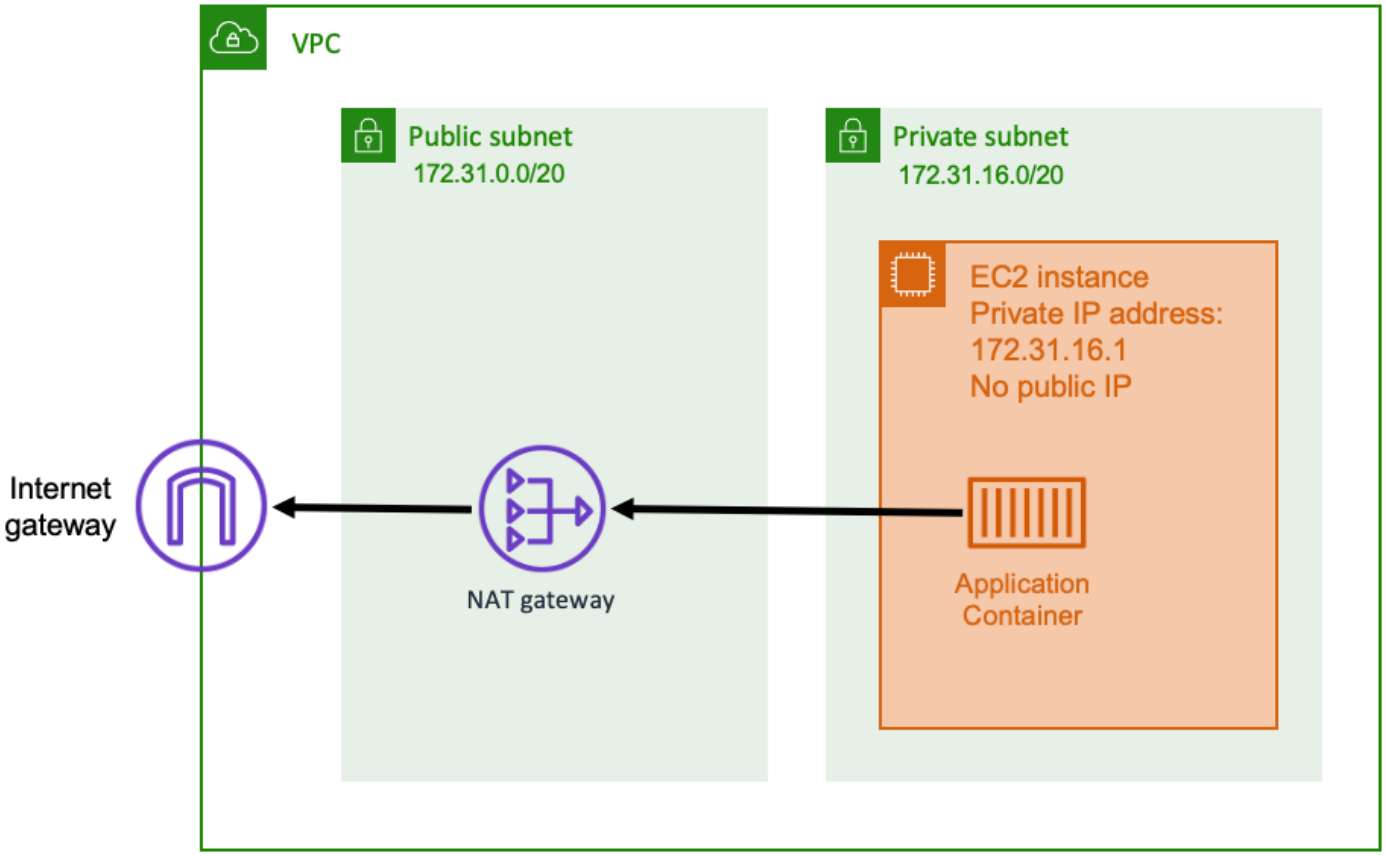
Par exemple, si votre application s'exécute sur Amazon EC2, assurez-vous que le port 22 pour l'accès SSH n'est pas ouvert. Dans le cas contraire, votre instance pourrait recevoir des tentatives de connexion SSH constantes de la part de robots malveillants sur Internet. Ces robots parcourent les adresses 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 des instances EC2 sur un sous-réseau public. Amazon ECS utilise ces instances EC2 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 aux ENI des tâches. 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 VPC 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 des instances EC2 sur un sous-réseau privé. Les conteneurs qui s'exécutent sur ces hôtes EC2 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.

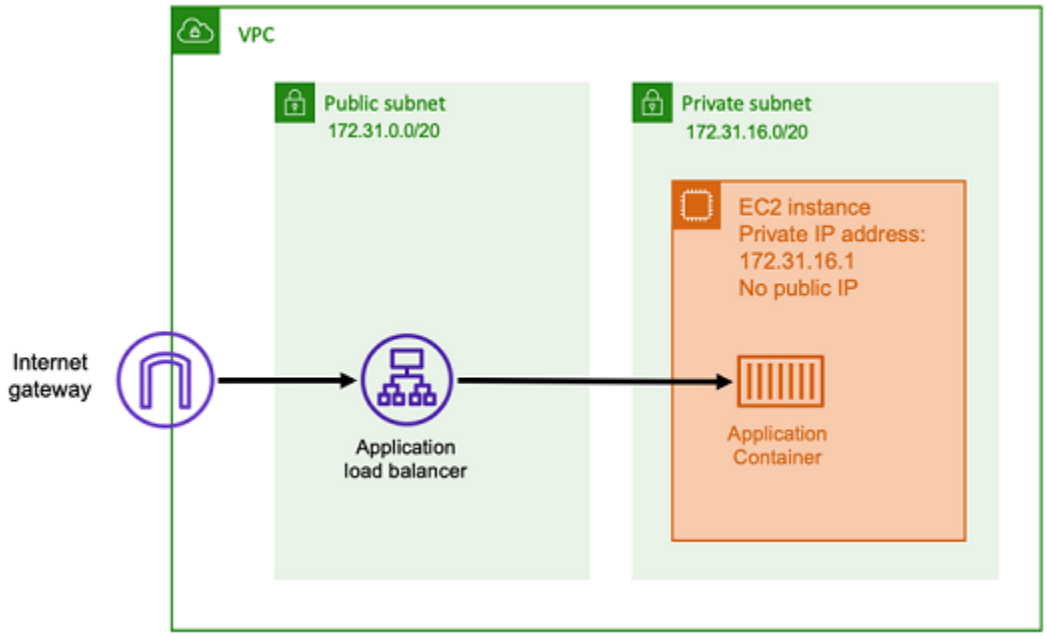
Bonnes pratiques pour recevoir des connexions entrantes vers Amazon ECS depuis Internet

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.

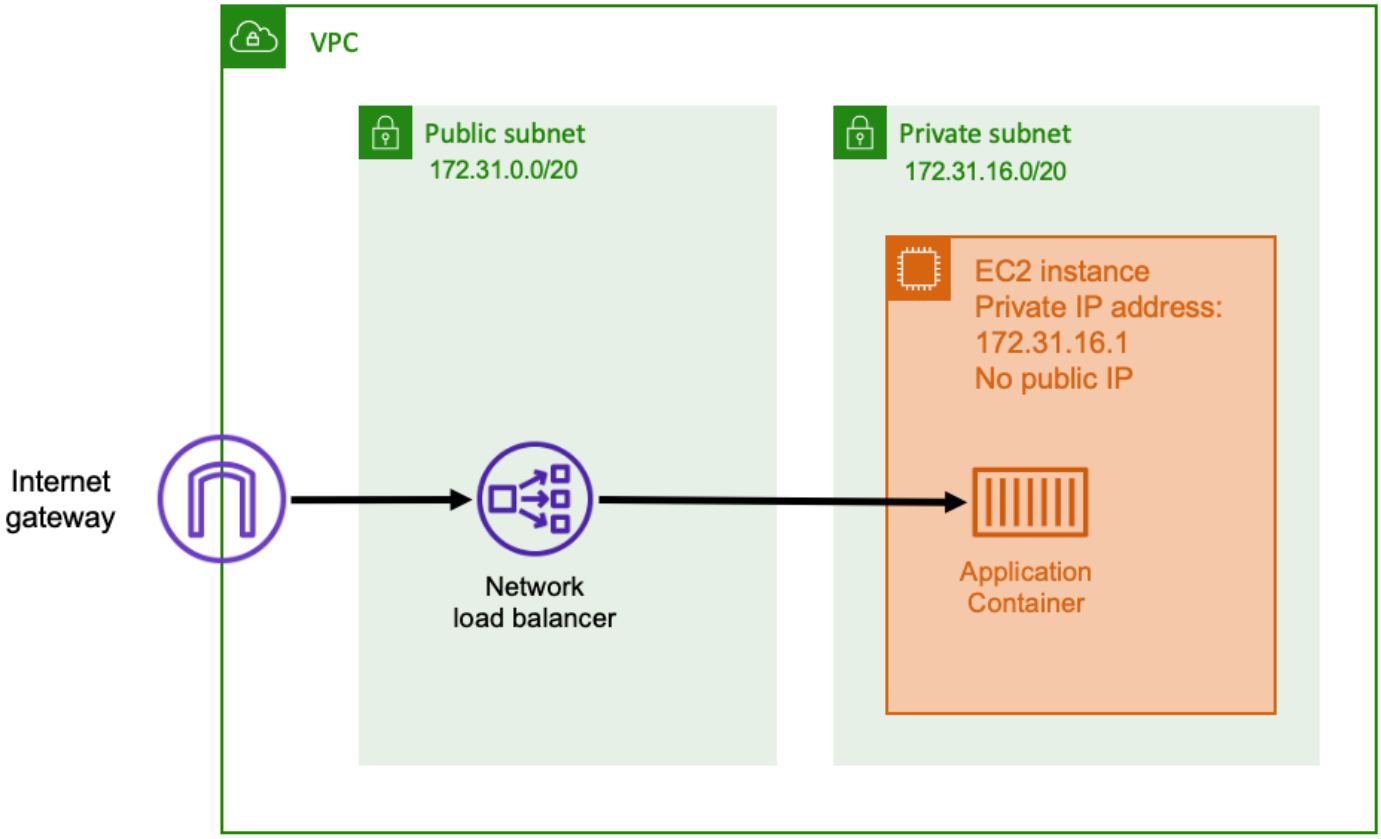
- Résiliation du protocole SSL/TLS : un Application Load Balancer peut garantir la sécurité des communications HTTPS et des certificats pour les communications avec les clients. Il peut éventuellement mettre fin à la connexion SSL au niveau de l'équilibreur de charge afin que vous n'ayez pas à gérer 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.

- nd-to-end Chiffrement E : 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

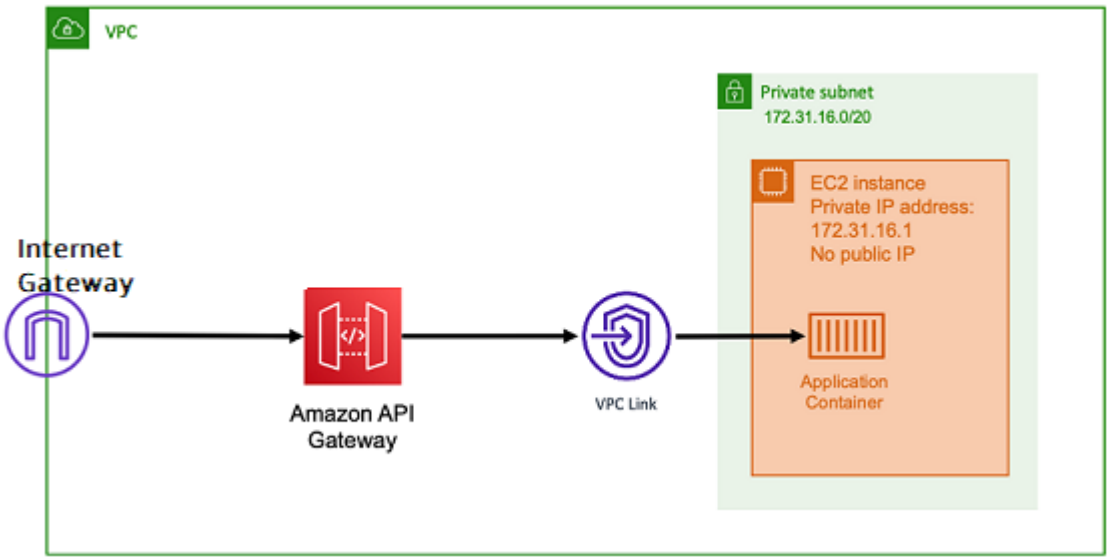
Comme le Network Load Balancer ne respecte pas le protocole d'application dans les couches supérieures du modèle OSI, il ne peut pas envoyer de messages de fermeture aux clients utilisant ces protocoles. Contrairement à l'Application Load Balancer, ces connexions doivent être fermées par l'application 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 la modification des demandes/réponses. 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 Notes de terrain : API basées sur des conteneurs sans serveur avec Amazon ECS et Amazon API Gateway.

Accès aux fonctionnalités d'Amazon ECS via 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.

Amazon Resource Names (ARN) et ID

Noms de ressources : `serviceLongArnFormat`, `taskLongArnFormat`, et `containerInstanceLongArnFormat`

Amazon ECS introduit un nouveau format pour les noms de ressources Amazon (ARN) et pour les ID de ressources pour les tâches, les instances de conteneur et les services 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. Pour plus d'informations, consultez Amazon Resource Names (ARN) et ID.

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.

AWSVPC tronquage

Nom de la ressource : `awsvpcTrunking`

Amazon ECS prend en charge le lancement d'instances de conteneur avec des limites d'interface réseau Elastic (ENI) augmentées à l'aide de types d'instances Amazon EC2 pris en charge. Lorsque vous utilisez ces types d'instances et adoptez le paramètre de compte `awsvpcTrunking`, des ENI supplémentaires sont disponibles sur les instances de conteneur nouvellement lancées. Vous pouvez utiliser cette configuration pour placer plus de tâches en

utilisant le mode réseau `awsvpc` sur chaque instance de conteneur. Grâce à cette fonctionnalité, une instance `c5.large` pour laquelle le paramètre `awsvpcTrunking` est activé possède une quota ENI de dix. L'instance de conteneur a alors une interface réseau principale. Amazon ECS crée et attache une interface réseau « de jonction » à l'instance de conteneur. L'interface réseau principale et l'interface réseau de jonction ne sont pas prises en compte dans la quota ENI. Par conséquent, vous pouvez utiliser cette configuration pour lancer dix tâches sur l'instance de conteneur au lieu des deux tâches actuellement possibles. Pour plus d'informations, consultez Jonction d'interface réseau Elastic.

L'argument par défaut est `disabled`.

Seules les ressources lancées après l'acceptation bénéficient de l'augmentation des limites ENI. Toutes les ressources existantes ne sont pas affectées. Pour appliquer l'augmentation des quotas ENI à une instance de conteneur, il est nécessaire de la vider et d'enregistrer une nouvelle instance de conteneur sur le cluster.

CloudWatch Informations sur les conteneurs

Nom de la ressource : `containerInsights`

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. Conteneur Insights fournit également des informations de diagnostic (par exemple sur les échecs de redémarrage des conteneurs) pour vous aider à isoler les problèmes et à les résoudre rapidement. Vous pouvez également définir des alarmes CloudWatch sur les métriques collectées par Container Insights. Pour plus d'informations, consultez Surveillez les conteneurs Amazon ECS à l'aide de Container Insights.

Lorsque vous activez le paramètre de compte `containerInsights`, tous les nouveaux clusters ont Container Insights activé par défaut. Vous pouvez désactiver ce paramètre pour des clusters spécifiques lorsque vous les créez. Vous pouvez également modifier ce paramètre à l'aide de l' UpdateClusterSettings API.

Pour des clusters contenant des tâches ou des services utilisant le type de lancement EC2, vos instances de conteneur doivent exécuter la version 1.29.0 ou ultérieure de l'agent Amazon ECS pour utiliser Container Insights. Pour plus d'informations, consultez Gestion des instances de conteneur Linux.

L'argument par défaut est `disabled`.

IPv6 de VPC à double pile

Nom de la ressource : `dualStackIPv6`

Amazon ECS prend en charge la fourniture de tâches avec une adresse IPv6 en plus de l'adresse IPv4 privée principale.

Pour que les tâches reçoivent une adresse IPv6, la tâche doit utiliser le mode de réseau `awsvpc`, doit être lancé dans un VPC configuré pour le mode de double pile et le paramètre de compte `dualStackIPv6` 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 lancement EC2 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 plus d'informations, consultez Modification des paramètres de compte.

Si vous aviez une tâche en cours d'exécution avec le mode réseau `awsvpc` dans un sous-réseau IPv6 activé entre le 1er octobre 2020 et le 2 novembre 2020, le paramètre de compte par défaut `dualStackIPv6` dans la région dans laquelle la tâche était en cours d'exécution 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`.

Conformité de Fargate à la norme FIPS-140

Nom de la ressource : `fargateFIPSMode`

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).

L'argument par défaut est `disabled`.

Vous devez activer la conformité à la norme FIPS-140. Pour plus d'informations, consultez 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 plus d'informations, consultez Modification des paramètres de compte.

Autorisations des ressources de balises

Nom de la ressource : `tagResourceAuthorization`

Certaines actions d'API Amazon ECS vous permettent de spécifier des balises lorsque vous créez la ressource.

Amazon ECS introduit l'autorisation de balisage pour la création de ressources. Les utilisateurs doivent disposer d'autorisations pour les actions qui créent une ressource, telles que`ecsCreateCluster`. Si des balises sont spécifiées dans l'action de création de ressources, octroie AWS une autorisation supplémentaire à l'`ecs:TagResource`action afin de 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 plus d'informations, consultez the section called "Baliser des ressources pendant la création".

Période d'attente pour retirer une tâche Fargate

Nom de la ressource : `fargateTaskRetirementWaitPeriod`

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.

Activation de la surveillance d'exécution

Nom de la ressource : `guardDutyActivate`

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.

Rubriques

- Amazon Resource Names (ARN) et ID
- Calendrier relatif au format ARN et de l'ID de ressource
- AWS Fargate Conformité à la norme fédérale de traitement de l'information (FIPS-140)
- Autorisation de balisage
- Chronologie des autorisations de balisage
- AWS Fargate temps d'attente pour la retraite des tâches
- Surveillance du temps d'exécution (GuardDuty intégration Amazon)
- Affichage de vos paramètres de compte à l'aide de la console
- Modification des paramètres de compte
- Restauration des paramètres par défaut du compte Amazon ECS
- Gestion des paramètres de compte à l'aide d' AWS CLI

Amazon Resource Names (ARN) et ID

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 gérer Amazon ECS, des ARN ou des ID de ressource sont requis 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.

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, l'option d'entrée ou de retrait du nouveau format n'affecte pas vos ID de ressource existants.

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 de compte.

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 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 ID plus courts pour les tâches et les instances de conteneur Amazon ECS. Le format d'ID de ressource actuel comporte 36 caractères. Le nouveau format d'ID comporte 32 caractères sans traits d'union. Pour de plus amples informations sur le nouveau format d'ID de ressource, consultez Modification des paramètres de compte.

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.

AWS Fargate Conformité à la norme fédérale de traitement de l'information (FIPS-140)

Vous devez activer la conformité à la norme Federal Information Processing Standard (FIPS-140) sur Fargate. Pour plus d'informations, consultez the section called “AWS Fargate Conformité à la norme FIPS-140”.

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 plus d'informations, consultez 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 plus d'informations, consultez 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 sur`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 le test de votre 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 plus d'informations, consultez AWS FAQ sur la maintenance des tâches Fargate.

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
```

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 les instances de conteneur Fargate et EC2 en AWS surveillant en permanence les journaux et l'activité 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"
    }
}
```

Affichage de vos paramètres de compte à l'aide de la console

Vous pouvez utiliser le AWS Management Console pour consulter les paramètres de votre compte.

> **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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 de compte

Vous pouvez utiliser le AWS Management Console pour modifier les paramètres de votre compte.

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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. Pour augmenter ou diminuer le nombre de tâches que vous pouvez exécuter en mode réseau awsvpc pour chaque instance EC2, sous AWSVPCTrunking, sélectionnez Trunking. AWSVPC
6. Pour utiliser ou arrêter d'utiliser CloudWatch Container Insights par défaut pour les clusters, sous CloudWatch Container Insights, sélectionnez ou désactivez CloudWatchContainer Insights.
7. Pour activer ou désactiver l'autorisation de balisage, sous Autorisation de balisage des ressources, sélectionnez ou désactivez l'autorisation de balisage des ressources.
8. Sélectionnez Enregistrer les modifications.
9. Dans l'écran de confirmation, choisissez Confirm (Confirmer) pour enregistrer la sélection.

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 de compte à l'aide d’ AWS CLI

Vous pouvez gérer les paramètres de votre compte à l'aide de l'API Amazon ECS AWS CLI ou des SDK. Les paramètres `fargateFIPSMode` et `dualStackIPv6` les paramètres du `fargateTaskRetirementWaitPeriod` compte ne peuvent être consultés ou modifiés qu'à l'aide de ces outils.

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.

- Writer-ECS AccountSetting ()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.

- Writer-ECS AccountSetting ()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.

- Writer-ECS AccountSetting ()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
```

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 de tâche.

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 de tâche
- Architecture de votre application
- Création d'une définition de tâche à l'aide de la console
- Mise à jour d'une définition de tâche à l'aide de la console
- Annulation de l'enregistrement d'une révision de définition de tâche à l'aide de la console
- Suppression de l'enregistrement d'une révision de définition de tâche à l'aide de la console
- Cas d'utilisation de définition de tâche
- Exemples de définitions de tâches

États de définition de tâche

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 Amazon ECS : la définition de la tâche est requise pour que la tâche reste saine.
- Déploiements et ensembles de tâches Amazon ECS : la définition des tâches est requise lorsqu'un événement de mise à l'échelle est initié 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 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.

Architecture de votre application

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 lancement EC2, vous choisissez également le type d'instance. Pour certains types d'instances, tels que le GPU, vous devez définir des paramètres supplémentaires. Pour plus d’informations, consultez Cas d'utilisation de définition de tâche.
- 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.
- Le 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 une instance EC2, 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, car vous pouvez mieux contrôler la façon dont vos applications communiquent entre elles et avec les autres services au sein de vos VPC.

  Pour les tâches exécutées sur Fargate, vous ne pouvez utiliser `awsvpc` que 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 de tâche.

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 mettre à l'échelle ces conteneurs indépendamment. Par exemple, il est peu probable qu'un site Web et une API nécessitent une montée en puissance au même rythme. À mesure que le trafic augmente, le nombre de conteneurs Web requis sera 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, considérez chaque révision de définition de tâche comme un instantané ponctuel des paramètres d'une image de conteneur particulière. De la même manière, le conteneur est un instantané de tout ce qui est nécessaire pour exécuter une version particulière du code de votre application.

  Assurez-vous qu'il existe 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 lui demander de 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. Cette recommandation doit être faite en parallèle avec notre recommandation visant à fournir à chaque composante 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 également 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

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.
- Simplifiez le traitement de 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

L'un des choix les plus importants à faire lors du déploiement de 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 à la mise à l'échelle et à la planification des capacités. Dans Amazon ECS, deux mesures de ressources sont utilisées pour déterminer la capacité : le processeur et la mémoire. 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 déclarer des réservations et des limites de ressources.

Lorsque vous déclarez une réservation, vous déclarez le minimum de ressources requis par une tâche. Votre tâche reçoit au moins le montant des ressources demandées. Votre application peut être en mesure d'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 instances Amazon EC2 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 tout en minimisant 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 entrants et sortants 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 travailleurs individuels ou les serveurs de base de données, la capacité disponible et les coûts constituent vos principales considérations. 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.

Mise en réseau des tâches pour les tâches sur les instances Amazon EC2

Le comportement de mise en réseau des tâches Amazon ECS hébergées sur des instances Amazon EC2 dépend du mode réseau défini dans la définition de 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 sur EC2	Description
`awsvpc`	Oui	Oui	La tâche reçoit sa propre interface réseau Elastic (ENI) et une adresse IPv4 privée principale. Cela confère à la tâche les mêmes propriétés de réseaux que les instances Amazon EC2.
`bridge`	Oui	Non	La tâche utilise le réseau virtuel intégré de Docker sous Linux, qui s'exécute à l'intérieur de chaque instance Amazon EC2 hébergean

Mode réseau	Conteneurs Linux sur EC2	Conteneurs Windows sur EC2	Description
			t 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 de conteneur directement à l'ENI de l'instance Amazon EC2 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. Dans ce mode, vous ne pouvez pas exécuter plusieurs tâches de la même définition de tâche sur une instance Amazon EC2 unique.
`none`	Oui	Non	La tâche n'a pas de connectivité réseau externe.
`default`	Non	Oui	La tâche utilise le réseau virtuel intégré de Docker sous Windows, qui s'exécute à l'intérieur de chaque instance Amazon EC2 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.

Rubriques

- Mode réseau awsvpc
- Mode Host
- Mode Bridge

Mode réseau **awsvpc**

Les fonctions de mise en réseau des tâches fournies par le mode réseau `awsvpc` attribuent aux tâches Amazon ECS les mêmes propriétés de mise en réseau que les instances Amazon EC2. L'utilisation du mode `awsvpc` réseau simplifie la mise en réseau des conteneurs, car vous pouvez mieux contrôler la façon dont vos applications communiquent entre elles et avec les autres services au sein de vos VPC. 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 réseau d'Amazon EC2, 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'instance Amazon EC2 hôte avec le groupe de sécurité spécifié. La tâche envoie et reçoit le trafic réseau sur l'ENI de la même manière que les instances Amazon EC2 avec leurs interfaces réseau principales. Chaque ENI de tâche se voit attribuer une adresse IPv4 privée par défaut. Si votre VPC est activé pour le mode double pile et que vous utilisez un sous-réseau avec un bloc CIDR IPv6, l'ENI de tâche recevra également une adresse IPv6. Chaque tâche ne peut avoir qu'une seule ENI.

Ces ENI sont visibles dans la console Amazon EC2 de votre compte. Votre compte ne peut pas détacher ou modifier les ENI. 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 plus d'informations, consultez 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 plus d'informations, consultez Configuration réseau. Les tâches sont placées sur des instances Amazon EC2 compatibles dans les mêmes zones de disponibilité que ces sous-réseaux et les groupes de sécurité spécifiés sont associés à l'ENI qui est allouée pour la tâche.

Considérations relatives à Linux

Tenez compte des éléments suivants lorsque vous utilisez le système d'exploitation Linux.

- 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 plus d'informations, consultez 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 Linux Amazon EC2 Linux nécessite une version `1.15.0` ou une version ultérieure de l'agent de conteneur pour exécuter des tâches qui utilisent le mode réseau `awsvpc`. 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.

- Les tâches Amazon ECS qui utilisent le mode réseau `awsvpc` reçoivent chacune leur propre interface réseau Elastic (ENI), qui est attachée à l'instance Amazon EC2 qui l'héberge. Il existe un quota par défaut pour le nombre d'interfaces réseau qui peuvent être attachées à une instance Amazon EC2 Linux. L'interface réseau principale est considérée comme une seule pour ce quota. Par exemple, par défaut, une instance `c5.large` peut avoir jusqu'à trois ENI pouvant lui être attachées. L'interface réseau principale de l'instance est considérée comme une interface réseau. Vous pouvez attacher deux ENI supplémentaires 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 de plus amples informations sur les limites d'ENI par défaut pour chaque type d'instance, veuillez consulter Adresses IP privées par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.
- Amazon ECS prend en charge le lancement d'instances Amazon EC2 Linux avec une augmentation de la densité ENI et utilisant des types d'instances pris en charge. Lorsque vous activez le paramètre de compte `awsvpcTrunking` et enregistrez les instances Linux Amazon EC2 à l'aide de ces types d'instance dans votre cluster, ces instances ont des quotas ENI plus élevés. 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 une densité ENI supérieure avec la fonctionnalité de jonction, vos instances Amazon EC2 doivent utiliser la version `1.28.1` ou 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 Accès aux fonctionnalités d'Amazon ECS via les paramètres du compte. Pour en savoir plus sur la jonction ENI, consultez Jonction d'interface réseau Elastic.
- Lors de l'hébergement de tâches qui utilisent le mode réseau `awsvpc` sur les instances Linux Amazon EC2, les ENI de votre tâche ne reçoivent pas d'adresses IP publiques. 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 les ENI qu'il attache à vos instances Amazon EC2 Linux. Si vous avez associé manuellement des ENI à vos instances, Amazon ECS peut tenter d'ajouter une tâche à une instance qui ne dispose pas de suffisamment de cartes 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 attacher manuellement des ENI à vos instances.

- Les instances Amazon EC2 Linux doivent être enregistrées avec la capacité `ecs.capability.task-eni` pour pouvoir faire l'objet d'un placement des tâches avec le mode réseau `awsvpc`. Les instances exécutant la version `1.15.0-4` ou une version ultérieure d'`ecs-init` sont automatiquement enregistrées avec cet attribut.
- Les ENI créées et attachées à vos instances Amazon EC2 Linux ne peuvent pas être détachées manuellement ni modifiées par votre compte. L'objectif est d'empêcher la suppression accidentelle de toute ENI associée à une tâche en cours d'exécution. Pour libérer les ENI d'une tâche, arrêtez cette dernière.
- 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`. En effet, les tâches qui utilisent le mode réseau `awsvpc` sont associées à une ENI, et non à une instance Amazon EC2 Linux. Pour plus d'informations, consultez Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge.
- 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 disposez d'un

cluster qui contient des AMI Windows Server 2016 optimisées pour Amazon ECS et des AMI Windows qui prennent en charge le mode réseau `awsvpc`, les tâches qui utilisent le mode réseau `awsvpc` ne sont pas lancées sur les instances Windows 2016 Server. Elles sont plutôt lancées sur des instances qui prennent en charge le mode réseau `awsvpc`.

- Votre instance Windows Amazon EC2 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 plus d'informations, consultez 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 qui utilisent le mode réseau `awsvpc`. Lorsque vous amorcez l'instance, vous devez configurer les options requises pour le mode réseau `awsvpc`. Pour plus d'informations, consultez 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.
- Les tâches Amazon ECS qui utilisent le mode réseau `awsvpc` reçoivent chacune leur propre interface réseau Elastic (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 attachées à une instance Amazon EC2 Windows. L'interface réseau principale est considérée comme une seule pour ce quota. Par exemple, par défaut, une instance `c5.large` peut avoir jusqu'à trois ENI qui lui sont attachées. L'interface réseau principale de l'instance compte parmi celles-ci. Vous pouvez attacher deux ENI supplémentaires 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 de plus amples informations sur les limites d'ENI par défaut pour chaque type d'instance, veuillez consulter Adresses IP privées par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.

- Lors de l'hébergement de tâches qui utilisent le mode réseau `awsvpc` sur les instances Windows Amazon EC2, les ENI de votre tâche ne reçoivent pas d'adresses IP publiques. 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 les ENI qu'il a attachées à votre instance Windows Amazon EC2. Si vous avez associé manuellement des ENI à vos instances, Amazon ECS peut tenter d'ajouter une tâche à une instance qui ne dispose pas de suffisamment de cartes 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 attacher manuellement des ENI à vos instances.
- Les instances Amazon EC2 Windows doivent être enregistrées avec l'interface réseau `ecs.capability.task-eni` pour pouvoir faire l'objet d'un placement des tâches avec le mode réseau `awsvpc`.
- Vous ne pouvez pas modifier ou détacher manuellement les ENI créées et attachées à vos instances Windows Amazon EC2. Cela vous évite de supprimer accidentellement une ENI associée à une tâche en cours d'exécution. Pour libérer les ENI d'une tâche, arrêtez cette dernière.
- 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`. En effet, les tâches qui utilisent le mode réseau `awsvpc` sont associées à une ENI, et non à une instance Amazon EC2 Windows. Pour plus d'informations, consultez Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge.

- 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 réseau `awsvpc` dans une configuration Windows EC2 :
  - 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 via IPv4 ou IPv6, ou les deux. Les adresses IPv4 et IPv6 sont indépendantes l'une de l'autre. 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 rubrique Migration vers IPv6 dans le Guide de l'utilisateur 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. De ce fait, les tâches auxquelles une adresse IPv6 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 adresse IPv6 est attribuée aux tâches Amazon ECS si les conditions suivantes sont réunies :

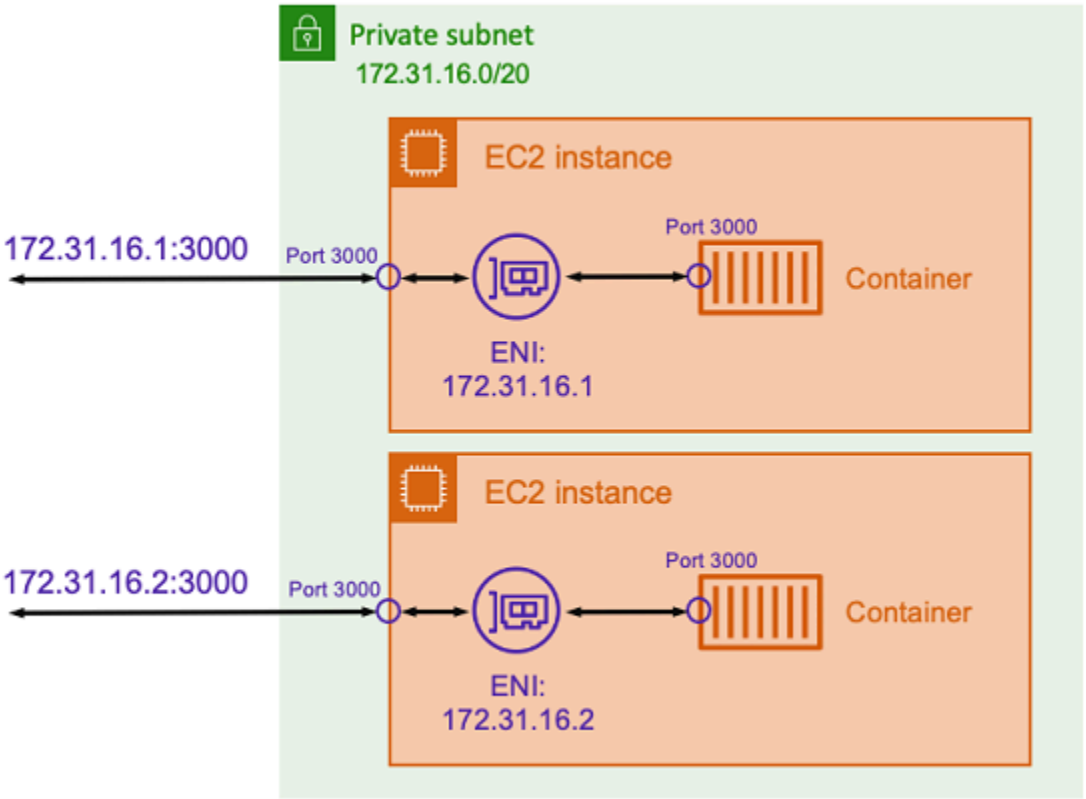
- L'instance Linux Amazon EC2 hébergeant la tâche utilise la version `1.45.0` ou version ultérieure de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de l'agent utilisée par votre instance et la mise à jour si nécessaire, veuillez consulter Mise à jour de l'agent de conteneur Amazon ECS.
- Le paramètre de compte `dualStackIPv6` est activé. Pour plus d'informations, consultez Accès aux fonctionnalités d'Amazon ECS via 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, veuillez consulter les rubriques Migration vers IPv6 et Modifier l'attribut d'adressage IPv6 de votre sous-réseau dans le Guide de l'utilisateur Amazon VPC.

Mode Host

Le mode réseau `host` est uniquement pris en charge pour les tâches Amazon ECS hébergées sur des instances Amazon EC2. 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 réseau `host` est utilisé, le conteneur reçoit le trafic sur le port 3 000 en utilisant l'adresse IP de l'instance Amazon EC2 de l'hôte sous-jacent. Nous vous recommandons de ne pas utiliser ce mode.

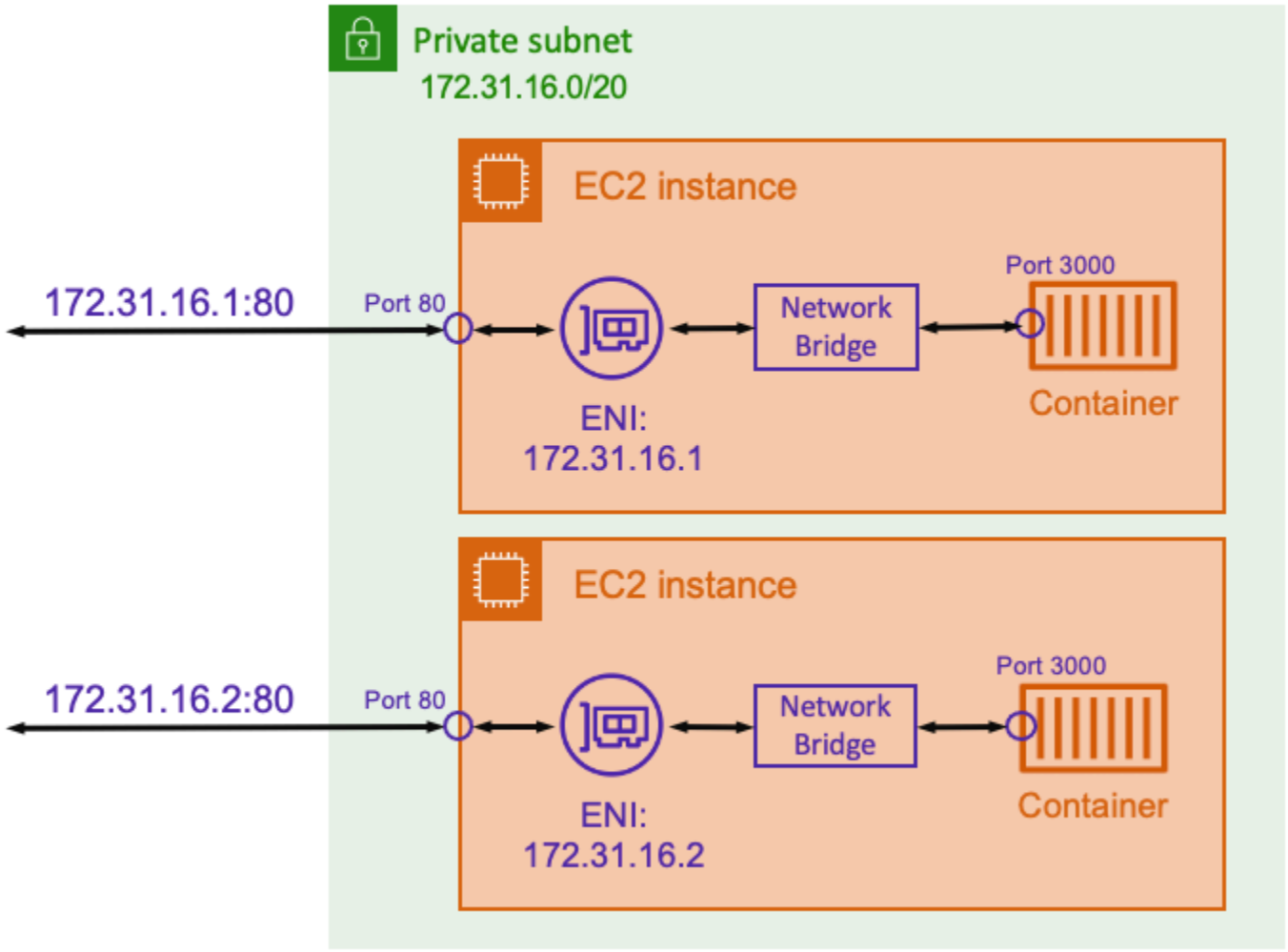
L'utilisation de ce mode réseau présente des inconvénients importants. Vous ne pouvez exécuter qu'une seule instanciation d'une tâche sur chaque hôte. Cela est dû au fait que seule la première tâche peut se lier au port requis sur l'instance Amazon EC2. Il est également impossible de remapper un port 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.

Mode Bridge

Le mode réseau `bridge` est uniquement pris en charge pour les tâches Amazon ECS hébergées sur des instances Amazon EC2.

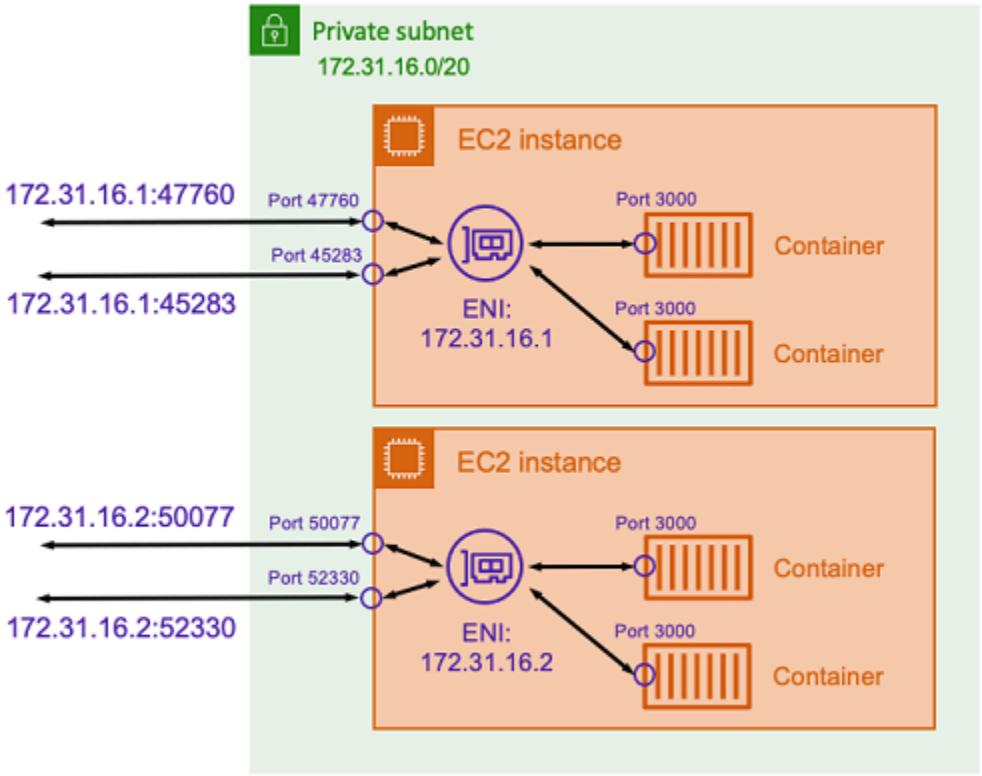
Avec le mode `bridge`, vous utilisez un pont réseau virtuel pour créer une couche entre l'hôte et la mise en réseau du conteneur. De cette façon, vous pouvez créer des mappages de ports qui remappent un port hôte à un port de conteneur. Les mappages peuvent être statiques ou dynamiques.



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'instance Amazon EC2. 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 sur le port `3000` du conteneur peut se voir attribuer un numéro de port aléatoire élevé, comme `47760`, sur l'hôte Amazon EC2. 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.

Mise en réseau des tâches pour les tâches sur 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 d'adresse CIDR IPv6, 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 l'utilisation des VPC et des sous-réseaux, veuillez consulter VPC et sous-réseaux 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 produit sur l'adresse IPv4 privée de la tâche. 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 aux API Amazon ECS 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 AWS PrivateLink dans le Guide des bonnes pratiques Amazon ECS.

Pour des exemples d'utilisation de la `NetworkConfiguration` ressource avec AWS CloudFormation, voirthe section called “Création de ressources Amazon ECS à l'aide de piles distinctes”.

Les ENI créées sont entièrement gérées 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
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

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
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 relatives à la mise en réseau des tâches Fargate

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 plus d’informations, consultez 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 les ENI créées et attachées par Fargate. L'objectif est d'empêcher la suppression accidentelle de toute ENI associée à une tâche en cours d'exécution. Pour libérer les ENI d'une tâche, arrêtez cette dernière.
- 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.
- Les tâches lancées dans des sous-réseaux avec des blocs CIDR IPv6 ne reçoivent qu'une adresse IPv6 lors de l'utilisation de la version `1.4.0` de la plateforme Fargate ou version ultérieure pour Linux ou `1.0.0` pour Windows.
- Pour les tâches utilisant la version `1.4.0` de la plateforme ou ultérieure pour Linux ou `1.0.0` pour Windows, les ENI de tâche prennent en charge les trames 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 plus d'informations, consultez Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge.

Utilisation d'un VPC en mode double pile

Lorsque vous utilisez un VPC en mode double pile, vos tâches peuvent communiquer via IPv4, IPv6 ou les deux. Les adresses IPv4 et IPv6 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, veuillez consulter la rubrique Migration vers IPv6 dans le Guide de l'utilisateur Amazon VPC.

Si les conditions suivantes sont réunies, des tâches Amazon ECS sur Fargate sont attribuées à une adresse 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 plus d’informations, consultez Accès aux fonctionnalités d'Amazon ECS via 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 rubrique Migration vers IPv6 dans le Guide de l'utilisateur 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, veuillez consulter Modifier l'attribut d'adressage IPv6 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 IPv6 est attribuée peuvent accéder à Internet. 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.

Stockage éphémère de la tâche Fargate

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 clé AWS de chiffrement propre.

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 plus d’informations, consultez Montages liés.

Utilisation de volumes de données dans des tâches

Amazon ECS prend en charge les options de volume de données suivantes pour les conteneurs :

- Volumes Amazon EBS : ces volumes fournissent un stockage par blocs rentable, durable et performant pour les charges de travail conteneurisées gourmandes en données. Vous pouvez configurer au maximum 1 volume EBS pour le rattacher à une tâche ECS autonome lorsque vous exécutez la tâche. Vous pouvez également configurer 1 volume EBS par tâche pour le rattacher à chaque tâche lancée via un service ECS lorsque vous créez ou mettez à jour le service. Les volumes Amazon EBS sont pris en charge pour les tâches Linux hébergées sur des instances Fargate ou Amazon EC2. Pour plus d’informations, consultez Volumes Amazon EBS.
- Stockage des tâches éphémères Fargate : par défaut, les tâches Amazon ECS hébergées sur Fargate à l'aide d'une version de plateforme `1.4.0` ou ultérieure reçoivent un minimum de 20

GiB 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.

- Volumes Amazon Elastic File System (Amazon EFS) : ces volumes fournissent un stockage de fichiers simple, évolutif et persistant 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 lorsque vous exécutez des tâches sur des instances Fargate ou Amazon EC2. Pour plus d’informations, consultez Volumes Amazon EFS.
- Volumes Amazon FSx for Windows File Server : ces volumes fournissent des serveurs de fichiers Windows entièrement gérés. Ces serveurs de fichiers sont basés sur un système de fichiers Windows. Lorsque vous utilisez FSx for Windows File Server avec Amazon ECS, vous pouvez allouer vos tâches Windows avec un stockage de fichiers permanent, distribué, partagé et statique. Pour plus d’informations, consultez Volumes FSx for Windows File Server.

  Les conteneurs Windows sur Fargate ne prennent pas en charge cette option.
- Volumes Docker : ces volumes sont des volumes gérés par Docker qui sont créés `/var/lib/docker/volumes` sur l'instance Amazon EC2 hôte. 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 instances Amazon EC2. 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. Pour plus d’informations, consultez Volumes Docker.
- Montages par liaison : ces volumes se composent d'un fichier ou d'un répertoire sur l'hôte, tel qu'une instance Amazon EC2AWS Fargate ou un répertoire monté dans un conteneur. Les volumes hôte de montage lié sont pris en charge lorsque vous exécutez des tâches sur des instances Fargate ou Amazon EC2. Les volumes hôtes de montage lié utilisent un stockage éphémère sur Fargate. La quantité de stockage éphémère varie selon les versions de la plateforme Fargate. Vous pouvez demander jusqu'à 200 gibioctets (GiB) de stockage éphémère sur la plateforme Fargate Linux version 1.4.0 ou ultérieure. Pour plus d’informations, consultez Montages liés.

Volumes Amazon EBS

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. Vous pouvez configurer au maximum 1 volume EBS pour le rattacher à une tâche ECS autonome lorsque vous exécutez la tâche. Vous pouvez également configurer 1 volume EBS par tâche pour le rattacher à chaque tâche lancée via un service ECS lorsque vous créez ou mettez à jour le service.

Les volumes Amazon EBS attachés aux tâches Amazon ECS sont gérés par ECS en votre nom et 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.

Pour surveiller les performances de votre volume, vous pouvez également utiliser les CloudWatch métriques Amazon EBS Amazon. Pour plus d'informations sur les métriques, consultez CloudWatch les métriques Amazon pour Amazon EBS dans le guide de l'utilisateur Amazon EC2.

Pour plus d'informations sur les volumes Amazon EBS, consultez les volumes Amazon EBS dans le guide de l'utilisateur Amazon EC2.

Régions AWS et zones de disponibilité pour les volumes Amazon EBS

Les volumes Amazon EBS peuvent être attachés aux tâches Amazon ECS de la manière suivante : Régions AWS

Nom de la région	Code région
US East (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

Nom de la région	Code région
Asie-Pacifique (Hyderabad)	ap-south-2
Asie-Pacifique (Jakarta)	ap-southeast-3
Asie-Pacifique (Mumbai)	ap-south-1
Asie-Pacifique (Osaka)	ap-northeast-3
Asie-Pacifique (Séoul)	ap-northeast-2
Asie-Pacifique (Singapour)	ap-southeast-1
Asie-Pacifique (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Canada (Central)	ca-central-1
Europe (Francfort)	eu-central-1
Europe (Irlande)	eu-west-1
Europe (Londres)	eu-west-2
Europe (Milan)	eu-south-1
Europe (Paris)	eu-west-3
Europe (Espagne)	eu-south-2
Europe (Stockholm)	eu-north-1
Europe (Zurich)	eu-central-2
Moyen-Orient (Bahreïn)	me-south-1
Moyen-Orient (EAU)	me-central-1
Amérique du Sud (São Paulo)	sa-east-1

> **Important**
>
> Vous ne pouvez pas configurer de volumes Amazon EBS pour les associer aux tâches Fargate Amazon ECS dans `euc1-az2` les zones de disponibilité et. `use1-az3`

Considérations relatives aux volumes Amazon EBS

Tenez compte des points suivants lorsque vous utilisez des volumes Amazon EBS :

- Les volumes Amazon EBS ne sont pris en charge que pour les tâches Linux hébergées sur Fargate et les tâches de type lancement EC2 hébergées sur des instances Linux Nitro basées sur des Amazon Machine Images (AMI) optimisées pour Amazon ECS.
- Pour les tâches hébergées sur Fargate, les volumes Amazon EBS sont pris en charge sur la version de `1.4.0` plate-forme ou ultérieure (Linux). Pour plus d'informations, consultez Versions de plateforme Linux Fargate.
- Pour les tâches hébergées sur des instances Linux Amazon EC2, les volumes Amazon EBS sont pris en charge sur une AMI optimisée pour ECS ou une version ultérieure. `20231219` Pour plus d'informations, consultez Extraction des métadonnées d'AMI optimisée pour Amazon ECS.
- 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 l'utilisateur Amazon EC2.
- 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 plus d'informations, consultez 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.
- Vous pouvez configurer les volumes Amazon EBS lors du déploiement uniquement pour les services qui utilisent le type de déploiement de mise à jour continue et la stratégie de planification Replica.

- Amazon ECS ajoute automatiquement les balises réservées `AmazonECSCreated` et`AmazonECSManaged`. 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 associer un volume Amazon EBS à une tâche de type lancement EC2 lancée sur une instance de conteneur basée sur le Xen système. Pour plus d'informations sur les types d'instances, consultez la section Types d'instances dans le guide de l'utilisateur Amazon EC2.
- Vous ne pouvez pas configurer de volumes Amazon EBS pour les associer à des tâches Amazon ECS en cours d'exécution. AWS Outposts

Chiffrement des données pour les volumes Amazon EBS

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

Vous pouvez configurer le chiffrement Amazon EBS par défaut afin que tous les nouveaux volumes créés et attachés à une tâche soient chiffrés à l'aide de la clé KMS que vous configurez 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 EC2.

Les volumes Amazon EBS attachés à des tâches peuvent être chiffrés en utilisant soit une valeur par défaut Clé gérée par AWS avec un alias`alias/aws/ebs`, soit une clé symétrique gérée par le client. 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.

Politique relative aux clés KMS gérée par le client

Pour utiliser votre clé gérée par le client pour chiffrer le volume EBS configuré pour être rattaché à votre tâche, 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 des exemples de déclarations de politique clés qui doivent être jointes à votre politique de clé KMS pour permettre à 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.

```
{
      "Sid": "ReadOnlyPermissions"
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:DescribeKey",
      "Resource":"*"
    },
    {
      "Sid": "DataKeyGenerationForAmazonEBS"
      "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"
        }
      }
    },
    {
      "Sid": "GrantCreationForAmazonEBS"
      "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 liés aux autorisations clés, consultez Résolution des problèmes de connexion aux volumes Amazon EBS.

Configuration des volumes Amazon EBS lors du déploiement

Pour configurer un volume Amazon EBS à associer à votre tâche, vous devez spécifier la configuration du point de montage dans la définition de votre tâche, nommer le volume et le `configuredAtLaunch` définir `true` sur. L'extrait de code JSON de définition de tâche suivant montre 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 de tâche. Pour utiliser cet exemple, remplacez *`user input placeholders`* par vos propres informations.

```
{
```

```
    "family": "mytaskdef",
    "containerDefinitions": [
        {
            "name": "container-using-ebs",
            "image": "amazonlinux:2",
            "entryPoint": [
                "sh",
                "-c"
            ],
            "command": [
                "ls -la /mount/ebs"
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEbsVolume",
                    "containerPath": "/mount/ebs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEBSVolume",
            "configuredAtLaunch": true
        }
    ]
}
```

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 utilisez la commande suivante. Pour utiliser cet exemple de commande, remplacez *user input placeholders* par vos propres informations.

```
aws ecs register-task-definition \
    --cli-input-json file://path_to_json_file/task-definition.json
```

Pour créer et enregistrer une définition de tâche à l'aide du AWS Management Console, voirCréation d'une définition de tâche à l'aide de la console.

Après avoir enregistré la définition de tâche, vous pouvez configurer le volume lors du déploiement à l' AWS Management Console aide des API Amazon ECS ou en transmettant un fichier JSON d'entrée avec 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.

La configuration d'un volume lors du déploiement vous permet de créer des définitions de tâches qui ne sont pas limitées à un type de volume 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 pour vos charges de travail de production par rapport à vos environnements de pré-production.

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",
                "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"
            }
        }
    ]
```

> **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.

Pour plus d'informations sur la façon de diagnostiquer et de résoudre les problèmes que vous pouvez rencontrer lorsque vous configurez un volume EBS pour le rattacher à une tâche, consultez Résolution des problèmes d'attachement de volume Amazon EBS. 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

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`, `sizeInGiB`, `encrypted`, et`kmsKeyId`. Enregistrez l'extrait suivant sous forme de fichier JSON. Pour utiliser cet exemple, remplacez *`user input placeholders`* par vos propres informations.

```
{
   "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"
            }
        }
    ]
}
```

La commande suivante peut être utilisée pour exécuter une tâche autonome à l'aide d'un fichier d'entrée JSON. La configuration spécifiée dans le fichier JSON est utilisée pour créer et associer un volume EBS à la tâche autonome. Pour utiliser cet exemple, remplacez *user input placeholders* par vos propres informations.

```
aws ecs run-task \
    --cli-input-json file://path_to_json_file/task.json
```

Pour configurer un volume en vue de le rattacher à une tâche autonome à l'aide du AWS Management Console, voirExécuter une application en tant que tâche Amazon ECS.

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 à l'aide du`snapshotId`. Enregistrez l'extrait suivant sous forme de fichier JSON. Pour utiliser cet exemple, remplacez *user input placeholders* par vos propres informations.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws::iam:1111222333:role/ecsInfrastructureRole",
              "snapshotId": "snap-12345"
            }
        }
   ]
}
```

Vous pouvez utiliser la commande suivante pour créer un service à l'aide d'un fichier d'entrée JSON. La configuration spécifiée dans le fichier JSON est utilisée pour créer et associer un volume EBS à chaque tâche gérée par le service. Pour utiliser cet exemple de commande, remplacez *user input placeholders* par vos propres informations.

```
aws ecs create-service \
    --cluster mycluster \
    --service-name mysvc \
    --cli-input-json file://path_to_json_file/create-service.json
```

Pour configurer un volume à associer à des tâches gérées par des services lors de la création de services à l'aide du AWS Management Console, voir. Création d'un service à l'aide de la console

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 définition de tâche `configuredAtLaunch` définie sur`true`. L'extrait de code JSON suivant montre la syntaxe permettant de configurer les `filesystemType` paramètres `volumeTypesizeInGiB`, `throughputiops`, et des volumes Amazon EBS configurés pour être joints à des tâches gérées par un service. Un volume est associé à chaque tâche du service. Pour utiliser cet exemple, remplacez *user input placeholders* par vos propres informations.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
  "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws::iam:1111222333:role/ecsInfrastructureRole",
               "volumeType": "gp3",
                "sizeInGiB": 100,
                 "iops": 3000,
                "throughput": 125,
                "filesystemType": "ext4"
            }
        }
   ]
```

```
}
```

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. Pour utiliser cet exemple, remplacez *`user input placeholders`* par vos propres informations.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": []
}
```

Vous pouvez utiliser la commande suivante pour mettre à jour un service à l'aide d'un fichier d'entrée JSON. Pour utiliser cet exemple de commande, remplacez *`user input placeholders`* par vos propres informations.

```
aws ecs update-service \
    --cli-input-json file://path_to_json_file/update-service.json
```

Pour configurer un volume à associer à des tâches gérées par le service lors de la mise à jour du service à l'aide du AWS Management Console, voir. Mise à jour d'un service à l'aide de la console

Politique de résiliation des 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.

Marquage des 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. Amazon ECS attache automatiquement les balises `AmazonECSCreated` et les balises `AmazonECSManaged` réservées à un volume Amazon EBS. 48 balises supplémentaires définies par l'utilisateur, gérées par ECS et propagées peuvent être ajoutées à un volume pour un total de 50 balises maximum par volume.

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`*`et``aws: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 à l'aide d'une balise définie par l'utilisateur et de balises gérées par ECS. 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"
          }
       }
   ]
}
```

> **Note**
>
> 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 (gp3ou) 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 vCPU)	IOPS Amazon EBS de référence (16 Kio d'E/S)	Débit Amazon EBS de référence (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

Unités de processeur demandées (en vCPU)	IOPS Amazon EBS de référence (16 Kio d'E/S)	Débit Amazon EBS de référence (entrée MiBps, 128 Kio d'E/S)	Bande passante de référence (en Mbits/s)
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.

Volumes Amazon EFS

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. Les sections suivantes vous décrivent comment commencer à utiliser Amazon EFS avec Amazon ECS.

Pour obtenir un didacticiel, consultez Utilisation des systèmes de fichiers Amazon EFS avec Amazon ECS à l'aide de la console.

Considérations relatives aux volumes Amazon EFS

Tenez compte des éléments suivants lorsque vous utilisez des volumes Amazon EFS :

- Pour les tâches utilisant le type de lancement EC2, la prise en charge du système de fichiers Amazon EFS a été ajoutée en tant que version préliminaire publique à l'AMI optimisée pour Amazon ECS version `20191212` et l'agent de conteneur version 1.35.0. 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 plus d'informations, consultez AMI optimisée pour Amazon ECS.

> **Note**
>
> Si vous créez votre propre AMI, vous devez utiliser l'agent de conteneur version 1.38.0 ou une version ultérieure, `ecs-init` version 1.38.0-1 ou une version 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 plus d'informations, consultez Versions de plateforme Linux Fargate.
- 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 de supervision utilise une faible proportion de la mémoire 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 lancement Amazon EC2, consultez. Point 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.

Utilisation des 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 tâche.

Spécification d'un système de fichiers Amazon EFS dans la définition de tâche

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": "amazonlinux:2",
            "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 : chaîne

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 : chaîne

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 : chaîne

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.

Volumes FSx for Windows File Server

FSx for 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 allouer vos tâches Windows avec un stockage de fichiers permanent, distribué, partagé et statique. Pour plus d'informations, veuillez consulter Qu'est-ce qu'Amazon FSx for Windows File Server ?.

> **Note**
>
> Les instances EC2 qui utilisent l'AMI complète Windows Server 2016 optimisée pour Amazon ECS ne prennent pas en charge FSx pour les volumes de tâches ECS FSx for Windows File Server.
>
> Vous ne pouvez pas utiliser de volumes FSx for Windows File Server dans un conteneur Windows sur une configuration Fargate.

Vous pouvez utiliser FSx for Windows File Server pour déployer des applications Windows nécessitant un accès au stockage externe partagé, au stockage régional hautement disponible ou au stockage haut débit. Vous pouvez monter un ou plusieurs volumes du système de fichiers FSx for Windows File Server sur un conteneur Amazon ECS qui s'exécute sur une instance Windows Amazon ECS. Vous pouvez partager les volumes du système de fichiers FSx for Windows File Server entre plusieurs conteneurs Amazon ECS au sein d'une même tâche Amazon ECS.

Pour activer l'utilisation de FSx for Windows File Server avec ECS, vous devez inclure l'identifiant de système de fichiers FSx for Windows File Server et les informations connexes 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 instance ECS Windows EC2 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 système de fichiers FSx for 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.

Les sections suivantes décrivent l'utilisation de FSx for Windows File Server avec Amazon ECS.

Pour voir un didacticiel connexe, consultez Utilisation de systèmes de fichiers FSx for Windows File Server avec Amazon ECS.

Considérations relatives aux volumes FSx for Windows File Server

Tenez compte des éléments suivants lorsque vous utilisez des volumes FSx for Windows File Server :

- FSx for Windows File Server avec Amazon ECS prend en charge uniquement les instances Amazon EC2 Windows. Les instances Amazon EC2 Linux ne sont pas prises en charge.
- FSx for Windows File Server avec Amazon ECS ne prend pas en charge AWS Fargate.
- FSx for Windows File Server avec Amazon ECS avec le mode réseau `awsvpc` nécessite la 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. Chaque tâche avec un volume FSx for Windows File Server reçoit une lettre de lecteur qui lui est attribuée.
- 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 plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.
- En général, les tâches s'exécutent uniquement sur le même VPC que le système de fichiers FSx for Windows File Server. Toutefois, il est possible de prendre en charge plusieurs VPC s'il existe une connectivité réseau établie entre le VPC de cluster Amazon ECS et le système de fichiers FSx for Windows File Server via l'appairage de VPC.
- Vous contrôlez l'accès à un système de fichiers FSx for Windows File Server au niveau du réseau en configurant les groupes de sécurité VPC. Seules les tâches hébergées sur des instances EC2 associées au domaine Active Directory avec des groupes de sécurité Active Directory correctement

configurés peuvent accéder au partage de fichiers FSx for Windows File Server. Si les groupes de sécurité sont mal configurés, ECS ne parvient pas à lancer la tâche et retourne le message d'erreur suivant : « `unable to mount file system` *`fs-id`*. »

- FSx for Windows File Server est intégré AWS Identity and Access Management à (IAM) pour contrôler les actions que vos utilisateurs et groupes IAM peuvent effectuer sur des ressources spécifiques de FSx for Windows File Server. Avec l'autorisation client, les clients peuvent définir des rôles IAM qui autorisent ou refusent l'accès à des systèmes de fichiers FSx for Windows File Server spécifiques, éventuellement exiger un accès en lecture seule et éventuellement autoriser ou interdire l'accès racine au système de fichiers à partir du client. Pour plus d'informations, veuillez consulter la rubrique Sécurité dans le Guide de l'utilisateur Amazon FSx Windows.

Spécification d'un système de fichiers FSx for Windows File Server dans la définition de tâche

Pour utiliser des volumes de système de fichiers FSx for 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 le système de fichiers FSx for Windows File Server pour le stockage des tâches.

`fileSystemId`

Type : chaîne

Obligatoire : oui

ID du système de fichiers FSx for Windows File Server à utiliser.

`rootDirectory`

Type : chaîne

Obligatoire : oui

Répertoire du système de fichiers FSx for Windows File Server à monter en tant que répertoire racine à l'intérieur de l'hôte.

`authorizationConfig`

`credentialsParameter`

Type : chaîne

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 : chaîne

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 EC2 auto-hébergé.

Méthodes de stockage des informations d'identification

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 paire clé/valeur, nom d'utilisateur/admin et mot de passe/*`mot de passe`*.

- 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 plus d'informations, consultez What is 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.

Volumes Docker

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, veuillez consulter Utilisation des volumes.

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 aux volumes Docker

Tenez compte des éléments suivants lorsque vous utilisez des volumes Docker :

- Les volumes Docker sont pris en charge uniquement en utilisant le type de lancement EC2 ou des 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
    ```

Spécification d'un volume Docker dans votre définition de tâche

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 instances EC2. 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 : chaîne

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 ou défini sur. `false`

`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. Pour plus d'informations, veuillez consulter Découverte de plug-ins Docker. Ce paramètre correspond à `Driver` dans la section Create a volume (Créer un volume) de Docker Remote API et mappe l'option `--driver` à `docker volume create`.

`driverOpts`

Type : chaîne

Obligatoire : non

Une carte des options spécifiques au pilote Docker à utiliser. Ce paramètre correspond à `DriverOpts` dans la section Create a volume (Créer un volume) de Docker Remote API et mappe l'option `--opt` à `docker volume create`.

`labels`

Type : chaîne

Obligatoire : non

Métadonnées personnalisées à ajouter à votre volume Docker. Ce paramètre correspond à `Labels` dans la section Create a volume (Créer un volume) de Docker Remote API et mappe l'option `--label` à `docker volume create`.

`mountPoints`

Type : tableau d'objets

Obligatoire : non

Les points de montage des volumes de données de votre conteneur. Ce paramètre correspond à `Volumes` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--volume` correspond à 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.

`sourceVolume`

Type : chaîne

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : chaîne

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`.

Exemples de volumes Docker

Voici des exemples illustrant l'utilisation des volumes Docker.

Pour fournir un stockage non permanent pour 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 permanent pour un conteneur 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"
               }
```

```
            ]
        }
    ]
```

Montages liés

Avec les montages par liaison, un fichier ou un répertoire sur un hôte, tel qu'une instance Amazon EC2, est monté dans un conteneur. Les montages liés sont pris en charge lorsque vous exécutez des tâches sur des instances Fargate ou 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 instances Amazon EC2, les données peuvent être liées au cycle de vie de l'instance Amazon EC2 hôte en spécifiant `host` une valeur `sourcePath` facultative dans votre définition de tâche. Pour de plus amples informations, veuillez consulter Utilisation de montages liés 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.

- Pour les tâches hébergées sur AWS Fargate une version de plate-forme `1.4.0` ou ultérieure (Linux) `1.0.0` ou ultérieure (Windows), elles reçoivent par défaut un minimum de 20 GiB de stockage éphémère pour les montages liés. Pour les tâches Linux, la quantité totale de stockage éphémère peut être augmentée, jusqu'à un maximum de 200 Gio, en spécifiant le paramètre `ephemeralStorage` dans votre définition de tâche. Par défaut, les tâches Amazon ECS Windows sur Fargate utilisant la version de plateforme `1.0.0` ou ultérieure reçoivent un minimum de 20 Gio de stockage éphémère. 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 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 instances Amazon EC2, lorsque les valeurs `host` et `sourcePath` ne sont pas spécifiées, le démon Docker gère le montage lié à votre place. 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 plus d'informations, consultez 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.

Spécification d'un montage lié dans votre définition de tâche

Pour les tâches Amazon ECS hébergées sur des instances Fargate ou Amazon EC2, l'extrait de code JSON de définition de tâche suivant indique la syntaxe des objets`mountPoints`, et pour une définition `volumes` de tâche. `ephemeralStorage`

```
{
   "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 instances Amazon EC2, vous pouvez utiliser le paramètre `host` 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 paramètre `host` est utilisé pour lier le cycle de vie du montage lié à l'instance Amazon EC2 hôte, plutôt qu'à la tâche et à l'endroit où elle est stockée. 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 est pris en charge uniquement lors de l'utilisation de tâches hébergées sur des instances Amazon EC2.

`sourcePath`

Type : chaîne

Obligatoire : non

Lorsque le paramètre `host` est utilisé, spécifiez un paramètre `sourcePath` pour déclarer le chemin d'accès sur l'instance Amazon EC2 hôte qui est présentée au conteneur. Si ce paramètre est vide, le démon Docker attribue un chemin hôte pour vous. Si le paramètre

`host` contient un emplacement de fichier `sourcePath`, le volume de données persiste à l'emplacement spécifié sur l'instance Amazon EC2 hôte jusqu'à ce que vous le supprimiez manuellement. Si la valeur `sourcePath` n'existe pas sur l'instance Amazon EC2 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 à `Volumes` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--volume` correspond à 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.

`sourceVolume`

Type : chaîne

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : chaîne

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`.

`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 lié

Les exemples suivants couvrent les cas d'utilisation les plus courants pour l'utilisation d'un montage lié 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 Amazon EC2.

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'instance Amazon EC2 hôte

Pour les tâches hébergées sur des instances Amazon EC2, vous pouvez utiliser des montages liés et lier les données au cycle de vie de l'instance Amazon EC2 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 valeur `containerPath` sont écrits dans la valeur `sourcePath` de l'instance Amazon EC2 hôte.

> **Important**
>
> Amazon ECS ne synchronise pas votre stockage entre les instances Amazon EC2. Les tâches qui utilisent un stockage permanent peuvent être placées sur n'importe quelle instance Amazon EC2 disposant de la capacité nécessaire dans votre cluster. Si vos tâches nécessitent un stockage permanent après l'arrêt et le redémarrage, spécifiez toujours la même instance Amazon EC2 au moment du lancement de la tâche à l'aide de AWS CLI la commande start-task. Vous pouvez également utiliser des volumes Amazon EFS pour le stockage permanent. Pour plus d'informations, consultez Volumes Amazon EFS.

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'instance Amazon EC2 hôte contient des données sous `/ecs/webdata` que vous souhaitez monter à l'intérieur du 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": "nginx",
    "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 instances Amazon EC2, vous pouvez définir un ou plusieurs volumes sur un conteneur, puis utiliser le paramètre `volumesFrom` dans une définition du conteneur différente, au sein de la même tâche, pour monter tous les volumes du `sourceContainer` à leurs

points de montage définis à l'origine. 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 d'échange de conteneur

Avec Amazon ECS, vous pouvez contrôler l'utilisation de l'espace mémoire d'échange sur vos instances Amazon EC2 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-memory erreurs 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 converti en l'option `--memory-swap` de la commande docker run, où la valeur est la somme de la mémoire du conteneur plus la valeur `maxSwap`.

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 est mappé à l'option `--memory-swappiness` de docker run.

Dans l'exemple suivant, la syntaxe JSON est fournie.

```
"containerDefinitions": [{
        ...
        "linuxParameters": {
            "maxSwap": integer,
            "swappiness": integer
        },
        ...
}]
```

Considérations relatives à l'échange de conteneur

Tenez compte des points suivants lorsque vous utilisez une configuration d'échange par conteneur.

- L'espace d'échange doit être activé et alloué sur l'instance Amazon EC2 hébergeant vos tâches pour que les conteneurs puissent l'utiliser. Par défaut, l'échange n'est pas activé pour les AMI optimisées Amazon ECS. Vous devez activer l'échange sur l'instance pour utiliser cette fonction. Pour plus d'informations, veuillez consulter la rubrique Volumes d'échange pour le stockage

d'instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux ou Comment allouer de la mémoire en tant qu'espace d'échange dans une instance Amazon EC2 en utilisant un fichier d'échange ?.

- Les paramètres de définition de conteneur d'espace d'échange sont uniquement pris en charge pour les définitions de tâche indiquant le type de lancement EC2. 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 de l'échange est limitée à deux fois la réservation de mémoire du conteneur.
- Si vous utilisez des tâches sur Amazon Linux 2023, le paramètre `swappiness` n'est pas pris en charge.

Considérations relatives à 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.

Rubriques

- Paramètres de définition de tâche
- Systèmes d'exploitation et architectures
- UC et mémoire au niveau de la tâche
- Mise en réseau des tâches
- Limites des ressources de tâche
- Journalisation
- Stockage de tâche
- Chargement différé d'images de conteneurs à l'aide de Seekable OCI (SOCI)

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 plus d’informations, consultez 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 plus d’informations, consultez Volumes.
- `cpu` - Pour les conteneurs Windows sur AWS Fargate, la valeur ne peut pas être inférieure à 1 vCPU.

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 Linux sur AWS Fargate, vous pouvez utiliser l'architecture de processeur X86_64 ou l'architecture ARM64 pour vos applications ARM. Pour plus d’informations, consultez the section called “Utilisation des charges de travail ARM 64 bits sur Amazon ECS”.

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. Bien que vous puissiez également spécifier l'UC et la mémoire au niveau du conteneur pour les tâches Fargate, cette opération est facultative. 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 les valeurs de mémoire dans le fichier JSON en MiB ou en Go. Vous pouvez spécifier les valeurs du processeur dans le fichier JSON sous la forme d'un entier exprimé en unités de processeur ou en processeurs virtuels (vCPU).

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)	Entre 32 Go et 120 Go par incréments de 8 Go	Linux

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
Note Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure		

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'adresse IPv4 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 `1024` 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 plus d’informations, consultez 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 les SDK AWS CLI et. 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.

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, consultezUtilisation du pilote du journal awslogs.

Pour plus d'informations sur l'utilisation du pilote de journal `awsfirelens` dans une définition de tâche, consultez Utilisation du routage personnalisé des journaux.

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 plus d'informations, consultez Volumes Amazon EBS.
- Volumes Amazon EFS pour le stockage permanent. Pour plus d'informations, consultez Volumes Amazon EFS.
- Supports de liaison pour un stockage éphémère. Pour plus d'informations, consultez Montages liés.

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 plateforme de Fargate et sur l'AMI optimisée pour Amazon ECS sur les instances Amazon EC2.

Index Seekable OCI

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.

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 tâches Linux avec l'architecture ARM64 ne prennent pas en charge le fournisseur de capacité Fargate Spot.
- Les images de conteneur incluses dans la définition de tâche doivent avoir des index SOCI dans le même registre de conteneurs que l'image.
- 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 la compression gzip 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.
- 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, supprimez l'index SOCI du registre des conteneurs. Si une image de conteneur dans la tâche ne répond pas à l'un des critères, cette image de conteneur est téléchargée selon la méthode par défaut.

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 plus d'informations, veuillez consulter Point de terminaison des métadonnées de tâches version 4 dans le Guide de l'utilisateur Amazon Elastic Container Service pour AWS Fargate(langue française non garantie).

Considérations relatives à EC2 Windows

Les tâches exécutées sur des instances Windows EC2 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 Windows Amazon EC2 :

- `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 Windows EC2 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 plus d'informations, consultez the section called “Configuration supplémentaire d'un rôle pour les tâches Windows”.

Création d'une définition de tâche à l'aide de la console

Pour faciliter au maximum la création de définitions de tâches, la console Amazon ECS propose des sélections par défaut pour de nombreux choix.

Vous pouvez créer une définition de tâche à l'aide de la console ou en modifiant 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 de`containerDefinitions`.

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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(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 plus d'informations, consultez 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 plus d'informations, consultez 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 indiquée en vCPU et la mémoire est indiquée en 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	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
plateforme Linux `1.4.0` ou ultérieure		
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 de processeur de tâche prises en charge sont comprises entre 128 unités de processeur (0,125 vCPU) et 10 240 unités de processeur (10 vCPU). 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. 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 instance Amazon EC2 prenant en charge les GPU possède une unité GPU pour chaque GPU. Pour plus d'informations, consultez the section called “Utilisation des GPU sur Amazon ECS”.
- 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, consultezUtiliser les paramètres de définition des tâches pour transmettre des variables d'environnement à un conteneur.

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 plus d'informations, consultez Utilisation du routage personnalisé des journaux.

   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. Pour plus d'informations, consultez Transférer des journaux vers un domaine Amazon OpenSearch Service.
   - 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	
Vérification de l'état Ce sont les commandes qui déterminent si un conteneur est sain. Pour plus d'informations, consultez 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	

Pour configurer cette option	Faites ceci	
	de l'état réussisse avant qu'elle ne soit considéré e 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 surveilla nce de l'état s'exécute nt. Les valeurs valides sont comprises entre 0 et 300. • Pour Retries (Tentativ es), entrez le nombre de tentatives d'exécuti on 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	
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 déterminent 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, sélectionnez Désactiver le réseau. • Pour configurer 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 à `WorkingDir` 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	
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'autresu limits, 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.	

Pour configurer cette option	Faites ceci
Ordre de démarrage des conteneurs 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.

k. (Facultatif) Choisissez Add more containers (Ajouter des conteneurs supplémentaires) pour ajouter des conteneurs supplémentaires à la définition de tâche.

11. (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.

12. (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 for 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 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. • Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume. b. Pour ajouter des points de montage supplémen	

Type de volume	Étapes	
	taires, 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 définitio n de tâche lors du montage du système de fichiers Amazon EFS, sélectionnez IAM authorization (Autorisa tion IAM). 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	
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	
Serveur de fichiers FSx for Windows	a. Dans ID de système de fichiers, choisissez l'ID de système de fichiers FSx for Windows File Server. b. Pour le répertoire racine, entrez le répertoire, entrez le répertoire dans le système de fichiers FSx for Windows File Server à monter en tant que répertoire racine sur l'hôte. c. Dans Credential parameter (Paramètre d’informations d’identification), choisissez la manière dont les informations d'identification 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. Dans le champ 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 EC2 auto-hébergé. 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	

Type de volume	Étapes	
	d'accès au conteneur pour monter le volume. • 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
Volume Amazon EBS	a. 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. • 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	

13. 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.

14. (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 plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez Autorisations IAM requises pour AWS Distro pour OpenTelemetry l'intégration à Amazon Managed Service for Prometheus.

15. (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.

16. Choisissez Créer pour enregistrer la définition de tâche.

Amazon ECS console JSON editor

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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.

Mise à jour d'une définition de tâche 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`

Amazon ECS console

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 plus d'informations, consultez Mise à jour d'un service à l'aide de la console.

Amazon ECS console JSON editor

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 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 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.

Annuler l'enregistrement d'une nouvelle définition de tâche (console Amazon ECS)

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 de l'enregistrement d'une révision de définition de tâche 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 Amazon ECS : la définition de la tâche est requise pour que la tâche reste saine.
- Déploiements et ensembles de tâches Amazon ECS : la définition des tâches est requise lorsqu'un événement de mise à l'échelle est initié 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 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.

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 plus d'informations, consultez 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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 Nom de la définition de la tâche, sélectionnez les révisions à supprimer, puis choisissez Actions, Supprimer.
6. Vérifiez les informations dans le champ de confirmation Supprimer, puis choisissez Supprimer pour terminer.

Cas d'utilisation de définition de tâche

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. De même, pour le type de lancement EC2, vous devez choisir des instances spécifiques conçues pour la charge de travail.

Rubriques

- Utilisation des GPU sur Amazon ECS
- Utilisation du transcodage vidéo sur Amazon ECS
- Utilisation de AWS Neuron sur Amazon Linux 2 sur Amazon ECS
- Utilisation d'instances DL1 Deep Learning sur Amazon ECS
- Utilisation des charges de travail ARM 64 bits sur Amazon ECS
- Utilisation du pilote du journal awslogs
- Utilisation du routage personnalisé des journaux
- Authentification de registre privé pour les tâches

- Utiliser les paramètres de définition des tâches pour transmettre des variables d'environnement à un conteneur
- Utiliser un fichier pour transmettre des variables d'environnement à un conteneur
- Transmission de données sensibles à un conteneur

Utilisation des GPU sur Amazon ECS

Amazon ECS supporte les charges de travail utilisant des GPU lorsque vous créez des clusters avec des instances de conteneurs prenant en charge les GPU. Les instances de conteneur GPU Amazon EC2 utilisant les types d'instances p2, p3, p5, g3, g4 et g5 fournissent un accès aux GPU NVIDIA. Pour plus d'informations, consultez Instances de calcul accélérées Linux du Guide de l'utilisateur Amazon EC2 pour les instances Linux.

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 plus d'informations, consultez AMI optimisée pour Amazon ECS.

Vous pouvez désigner plusieurs GPU dans votre définition de tâche pour le placement des tâches au niveau d'un conteneur. Amazon ECS planifie les instances de conteneurs disponibles qui prennent en charge les GPU et associe les GPU physiques aux conteneurs appropriés pour des performances optimales.

Les types d'instances GPU Amazon EC2 suivants sont pris en charge. Pour plus d'informations, veuillez consulter Instances Amazon EC2 P2, Instances Amazon EC2 P3, Instances P4d Amazon EC2, Instances P5 Amazon EC2 (langue française non garantie), Instances G3 Amazon EC2, Instances G4 d'Amazon EC2, et Instances G5 Amazon EC2.

Type d'instance	GPU	Mémoire GPU (Gio)	vCPU	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

Type d'instance	GPU	Mémoire GPU (Gio)	vCPU	Mémoire (Gio)
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
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

Vous pouvez récupérer l'identifiant Amazon Machine Image (AMI) pour les AMI optimisées pour Amazon ECS en interrogeant l'API AWS Systems Manager Parameter Store. Grâce à ce paramètre, vous n'avez pas besoin de rechercher manuellement les ID d'AMI optimisées pour Amazon ECS. 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. Les types de familles d'instances g2 ne sont pris en charge que sur les versions antérieures à la version `20230906` des AMI optimisées pour le GPU Amazon ECS.
> 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 pilotes NVIDIA/CUDA 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 ce qui suit avant de commencer à utiliser des GPU sur 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 plus d'informations, consultez Enregistrement d'une instance externe dans un cluster.
- Vous devez définir `ECS_ENABLE_GPU_SUPPORT` sur `true` dans votre fichier de configuration d'agent. Pour plus d'informations, consultez 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 plus d'informations, consultez 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 variable d'environnement `NVIDIA_VISIBLE_DEVICES` en tant que liste des ID de périphérique GPU assignés au conteneur par Amazon ECS. 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 plus d'informations, consultez AMI optimisée pour Amazon ECS. 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 soit utiliser la console Amazon EC2 AWS CLI, soit l'API et enregistrer manuellement les instances dans votre cluster.
- Le type d'instance p4d.24xlarge ne fonctionne qu'avec CUDA 11 ou version ultérieure.
- L'AMI optimisée pour le GPU Amazon ECS a IPv6 activé, ce qui provoque des problèmes lors de l'utilisation de `yum`. Cela peut être résolu en configurant `yum` pour 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 base NVIDIA/CUDA, vous devez définir la variable d'exécution de conteneur `NVIDIA_DRIVER_CAPABILITIES` sur l'une des valeurs suivantes :
  - `utility,compute`
  - `all`

  Pour savoir comment définir la variable, veuillez consulter Contrôle de l'exécution du conteneur NVIDIA sur le site Web de NVIDIA.
- Les GPU ne sont pas pris en charge sur les conteneurs Windows.

Lancer une instance de conteneur GPU

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 l'utilisateur Amazon EC2 pour les instances Linux.

  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 dans le guide de l'utilisateur Amazon EC2 pour les instances Linux.

  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 :

   - Remplacez le *`sous-réseau`* par l'ID du sous-réseau privé ou public dans lequel votre instance sera lancée.
   - Remplacez *`gpu_ami`* par l'ID AMI de l'étape précédente.
   - Remplacez *`t3.large`* par le type d'instance que vous souhaitez utiliser.
   - Remplacez *`region`* par le code de la 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.

- Remplacez *region* par le code de votre région.

```
aws ecs list-container-instances --cluster cluster-name --region region
```

Spécification de plusieurs GPU dans votre définition de tâche

Pour utiliser plusieurs GPU sur une instance de conteneur et l'exécution du GPU de Docker, assurez-vous de désigner le nombre de GPU nécessaires à votre conteneur dans la définition de tâche. Une fois les conteneurs supportant les GPU placés, l'agent de conteneur Amazon ECS épinglera le nombre souhaité de GPU physiques au bon conteneur. Le nombre de GPU réservés pour tous les conteneurs dans une tâche ne peut pas dépasser le nombre de GPU disponibles sur l'instance de conteneur sur laquelle la tâche est lancée. Pour plus d'informations, consultez Création d'une définition de tâche 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 utilise 2 GPU, exécute l'utilitaire `nvidia-smi`, 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"
}
```

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, veuillez consulter Utiliser les données utilisateur d'instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

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 AMI EC2 privée compatible avec 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
   ```

Utilisation du transcodage vidéo sur Amazon ECS

Pour utiliser des charges de travail de transcodage vidéo sur Amazon ECS, enregistrez des 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. Les instances Amazon EC2 VT1 utilisent des cartes de transcodage multimédia Xilinx U30 pour accélérer les charges de travail de transcodage vidéo en direct et prérendues.

> **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 à déployer VT1 sur Amazon ECS, prenez en compte ce qui suit :

- Vos clusters peuvent contenir une combinaison d'instances VT1 et non VT1.
- 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 avoir de performances améliorées sur les instances VT1.

- 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 que de cartes pour chacune de vos instances VT1.
- 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 permet de vous assurer que vous utilisez efficacement vos ressources et que vos tâches pour vos charges de travail de transcodage vidéo se trouvent sur vos instances VT1. Pour plus d'informations, consultez 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 plus d'informations, consultez Exigences relatives à la définition de tâche.

Utilisation d'une AMI VT1

Vous avez deux options pour exécuter une AMI sur des instances de conteneur Amazon EC2 pour 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 des AMI 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	vCPU	RAM (Gio)	Cartes accélératrices U30	Appareils SoC XCU30 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/dri/renderD129 ,/dev/dri/renderD13`

Type d'instance	vCPU	RAM (Gio)	Cartes accélérat rices U30	Appareils SoC XCU30 adressables	Chemins de l'appareil
					`0 ,/dev/ dri/ renderD13 1`

Type d'instance	vCPU	RAM (Gio)	Cartes accélérat rices U30	Appareils SoC XCU30 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/renderD13`

Type d'instance	vCPU	RAM (Gio)	Cartes accélérat rices U30	Appareils SoC XCU30 adressables	Chemins de l'appareil
					8 ,/dev/dri/renderD139 ,/dev/dri/renderD140 ,/dev/dri/renderD141 ,/dev/dri/renderD142 ,/dev/dri/renderD143

> **Important**
>
> Si la définition de tâche répertorie les périphériques dont l'instance EC2 ne dispose 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`.

Dans l'exemple suivant, la syntaxe utilisée pour définir une 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
```

Utilisation de AWS Neuron sur Amazon Linux 2 sur Amazon ECS

Vous pouvez enregistrer les instances Amazon EC2 Trn1, Amazon EC2 Inf1 et Amazon EC2 Inf2 dans vos clusters pour les charges de travail de machine learning.

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 Inf1 et Inf2 d'Amazon EC2 sont alimentées par des puces AWS Inferentia. Elles fournissent des performances élevées et les inférences de coûts les plus bas 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 plus d'informations, consultez 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. Vous configurez plutôt 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 plus d'informations, consultez Exigences relatives à la définition de tâche.

Utilisation de 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 Trn1, Inf1 et Inf2 Amazon EC2.

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 Deep Learning AMI 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	vCPU	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/neuron10 ,/dev/neuron11 ,/dev/neuron12 ,/dev/neuron13 ,/dev/`

Type d'instance	vCPU	RAM (Gio)	AWS puces accélératrices ML	Chemins de l'appareil
				`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	vCPU	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	vCPU	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`

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://your-bucket-of-models/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"
        }
    ]
}
```

Utilisation d'instances DL1 Deep Learning sur Amazon ECS

Pour utiliser des charges de travail Deep Learning sur Amazon ECS, enregistrez des instances Amazon EC2 DL1 dans vos clusters. Les instances Amazon EC2 DL1 sont alimentées par des 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 à déployer DL1 sur Amazon ECS, prenez en compte ce qui suit :

- Vos clusters peuvent contenir une combinaison d'instances DL1 et non DL1.
- 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 vos tâches pour les charges de travail de deep learning se trouvent sur vos instances DL1. Pour plus d'informations, consultez 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 AMI DL1

Vous avez 3 options pour exécuter une AMI sur des instances Amazon EC2 DL1 pour Amazon ECS :

- AWS Marketplace Les AMI fournies par Habana ici.
- Il s'agit des AMI de deep learning Habana fournies 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.

Exigences relatives à la définition de tâche

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 de conteneurs Linux sur Amazon EC2 qui affiche 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"
        }
    ]
}
```

Utilisation des charges de travail ARM 64 bits sur Amazon ECS

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 Graviton2. 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 de lancements Fargate ou EC2.
- Les tâches Linux avec l'architecture ARM64 ne prennent pas en charge le fournisseur de capacité Fargate Spot.
- 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 lancement Amazon EC2, consultez les points suivants pour vérifier que votre Région prend en charge le type d'instance que vous souhaitez utiliser :
  - Instances M6g Amazon EC2
  - Instances T4g Amazon EC2
  - Instances C6g Amazon EC2
  - Instances R6gd Amazon EC2
  - Instances X2gd Amazon EC2

  Vous pouvez également utiliser la commande `describe-instance-type-offerings` Amazon EC2 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 votre définition de tâche

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": "arm64v8/busybox",
        "cpu": 100,
        "memory": 100,
        "essential": true,
        "command": [ "echo hello world" ],
        "entryPoint": [ "sh", "-c" ]
    }
 ],
 "requiresCompatibilities": [ "FARGATE" ],
 "cpu": "256",
```

```
 "memory": "512",
 "runtimePlatform": {
        "operatingSystemFamily": "LINUX",
        "cpuArchitecture": "ARM64"
  },
 "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole"
}
```

Interfaces de configuration d'ARM

Vous pouvez configurer l'architecture du processeur ARM pour les définitions de tâche Amazon ECS à l'aide de l'une des interfaces suivantes :

- Console Amazon ECS
- AWS Command Line Interface (AWS CLI)
- AWS SDK
- AWS Copilote

Utilisation du pilote du journal awslogs

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 lancement EC2, vous pouvez afficher différents journaux de vos conteneurs dans un emplacement pratique, et cela empêche vos journaux de conteneur d'occuper de l'espace disque sur vos instances de conteneur. Cette rubrique explique comment vous pouvez commencer à utiliser le pilote de journal `awslogs` dans vos définitions de tâche.

> **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.

Activer le pilote de journal awslogs pour vos conteneurs

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 plus d'informations, consultez Spécification d'une configuration de journal dans votre définition de tâche.

Si vous utilisez le type de lancement EC2 pour vos tâches et que vous souhaitez activer le pilote de journal `awslogs`, 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**
>
> Si vous n'utilisez pas l'AMI optimisée pour Amazon ECS (avec au minimum la version 1.9.0-1 du package `ecs-init`) pour vos instances de conteneur, vous devez également spécifier que le pilote du journaux `awslogs` est disponible sur l'instance de conteneur lorsque vous démarrez l'agent à l'aide de la variable d'environnement suivante dans votre déclaration docker run ou dans votre fichier de variable d'environnement. Pour plus d'informations, consultez Installation de l'agent de conteneur Amazon ECS.
>
> ```
> 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.

Création d'un groupe de journaux

Le pilote de `awslogs` journal peut envoyer des flux de journaux à un groupe de CloudWatch journaux existant dans Logs ou créer un nouveau groupe de journaux en votre nom. AWS Management Console Fournit une option de configuration automatique, qui crée un groupe de journaux en votre nom en utilisant le nom de famille de définitions de tâches avec `ecs` comme préfixe. Sinon, vous pouvez définir manuellement vos options de configuration de journaux et spécifier l'option `awslogs-create-group` sur la valeur `true` pour créer les groupes de journaux en votre nom.

> **Note**
>
> Pour que l'option `awslogs-create-group` crée votre groupe de journaux, votre politique de rôle IAM d'exécution de tâche ou politique de rôle d'instance EC2 doit inclure l'autorisation `logs:CreateLogGroup`.

Le code suivant montre comment définir l'opération `awslogs-create-group`.

```
{
    "containerDefinitions": [
        {
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "firelens-container",
                    "awslogs-region": "us-west-2",
                    "awslogs-create-group": "true",
                    "awslogs-stream-prefix": "firelens"
                }
            }
    ]
}
```

Utilisation de la fonction de configuration automatique pour créer un groupe de journaux

Lorsque vous enregistrez une définition de tâche, dans la console Amazon ECS, vous pouvez autoriser Amazon ECS à configurer automatiquement vos CloudWatch journaux. Cela entraîne la création d'un groupe de journaux en votre nom en utilisant le nom de famille de la définition de tâche avec `ecs` comme préfixe. Pour plus d'informations, consultez the section called “Création d'une définition de tâche à l'aide de la console”.

Options du pilote du journal awslogs disponibles

Le pilote de journal `awslogs` prend en charge les options suivantes dans les définitions de tâches Amazon ECS. Pour plus d'informations, consultez la section Pilote de journalisation CloudWatch des journaux.

`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. Pour plus d'informations, consultez Création d'un groupe de journaux.

`awslogs-stream-prefix`

Obligatoire : Oui, lorsque vous utilisez le type de lancement Fargate. Facultatif pour le type de lancement EC2, obligatoire pour le type de lancement Fargate.

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 plus d'informations, consultez 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 plus d'informations, consultez 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.

`mode`

Obligatoire : non

Valeurs valides : `non-blocking` | `blocking`

Valeur par défaut : `blocking`

Cette option définit le mode de livraison des messages de journal du conteneur à CloudWatch Logs. Le mode de livraison que vous choisissez affecte la disponibilité des applications lorsque le flux de logs entre le conteneur et le conteneur CloudWatch est interrompu.

Si vous utilisez le `blocking` mode par défaut et que le flux de logs CloudWatch 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 à CloudWatch. Nous vous recommandons d'utiliser ce mode si vous souhaitez garantir la disponibilité du service et si vous acceptez une certaine perte de journaux.

`max-buffer-size`

Obligatoire : non

Valeur par défaut : `1m`

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.

Spécification d'une configuration de journal dans votre définition de tâche

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. Cette section décrit la configuration de journal nécessaire pour qu'un conteneur utilise le pilote du journal `awslogs`. Pour plus d'informations, consultez Création d'une définition de tâche à l'aide de la console.

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": "wordpress",
            "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": "mysql",
            "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"
}
```

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 Exécuter une application en tant que tâche Amazon ECS et Création d'un service à l'aide de la console.

Utilisation du routage personnalisé des journaux

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.

Considérations

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 sur Amazon EC2 sous 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.

- FireLenspour Amazon ECS est pris en charge dans les AWS CloudFormation modèles. 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 réseau `host`, il s'agit du groupe de sécurité associé à l'instance Amazon EC2 qui héberge 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 plus d’informations, consultez 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 de définition du cluster et de la tâche et l'Amazon Resource Name (ARN) du cluster en tant que clés de métadonnées à vos journaux de conteneur stdout/stderr. 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"
}
```

Autorisations IAM requises

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.

L'exemple de politique IAM suivant ajoute les autorisations requises pour le routage des journaux vers Firehose.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "firehose:PutRecordBatch"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

L'exemple de politique IAM suivant ajoute les autorisations requises pour le routage des journaux vers Amazon CloudWatch Logs.

```
{
 "Version": "2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Action": [
   "logs:CreateLogStream",
   "logs:CreateLogGroup",
   "logs:PutLogEvents"
  ],
  "Resource": "*"
 }]
}
```

Votre tâche peut également nécessiter le rôle d'exécution de tâche Amazon ECS dans les conditions suivantes. Pour plus d'informations, consultez 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.
- Si vous spécifiez un fichier de configuration personnalisé hébergé dans Amazon S3, votre rôle IAM d'exécution de tâche doit inclure l'autorisation `s3:GetObject` pour le fichier de configuration et l'autorisation `s3:GetBucketLocation` sur le compartiment Amazon S3 dans lequel se trouve le fichier. Pour de plus amples informations, veuillez consulter Spécifier des autorisations dans une politique dans le Guide de l'utilisateur Amazon Simple Storage Service.

  L'exemple de politique IAM suivant ajoute les autorisations requises pour récupérer un fichier à partir d'Amazon S3. Spécifiez le nom de votre compartiment Amazon S3 et le nom du fichier de configuration.

  ```
  {
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "s3:GetObject"
        ],
  ```

```
      "Resource": [
        "arn:aws:s3:::examplebucket/folder_name/config_file_name"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket"
      ]
    }
  ]
}
```

Limite de tampons Fluentd

Lorsque vous créez une définition de tâche, vous pouvez spécifier le nombre d'événements mis en mémoire tampon en spécifiant la valeur (en octets) dans `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 rejeter les messages tampons afin d'en ajouter de nouveaux. Les journaux perdus peuvent rendre difficile le dépannage. Définir la limite de la mémoire tampon peut éviter ce problème.

Ce qui suit montre la syntaxe permettant de spécifier le`log-driver-buffer-limit`. Remplacez `my_service_` par le nom de votre service. :

```
{
    "containerDefinitions": [
        {
            "essential": true,
            "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-
bit:stable",
            "name": "my_service_log_router",
            "firelensConfiguration": {
                "type": "fluentbit"
            },
            "logConfiguration": {
```

```
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "firelens-container",
                    "awslogs-region": "us-west-2",
                    "awslogs-create-group": "true",
                    "awslogs-stream-prefix": "firelens"
                }
            },
            "memoryReservation": 50
        },
        {
            "essential": true,
            "image": "httpd",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "firehose",
                    "region": "us-west-2",
                    "delivery_stream": "my-stream",
                    "log-driver-buffer-limit": "2097152"
                }
            },
            "dependsOn": [
                {
                    "containerName": "log_router",
                    "condition": "START"
                }
            ],
            "memoryReservation": 100
        }
    ]
}
```

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 lancement Amazon EC2 et le type de lancement Fargate avec la version `1.4.0` de la plateforme ou version ultérieure.
- L'option n'est valide que lorsque `logDriver` est défini sur `awsfirelens`.
- La limite de tampon par défaut est de 1 Mio.
- Les valeurs valides sont `0` et `536870912` (512 Mio).

- 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 limite de mémoire tampon. La quantité totale de mémoire tampon spécifiée doit être inférieure à `536870912` (512 Mio) lorsque vous ne spécifiez pas les valeurs `memory` et `memoryReservation` du conteneur. Plus précisément, vous pouvez avoir un conteneur d'applications avec le pilote de journal `awsfirelens` et l'option `log-driver-buffer-limit` définie sur 300 Mio. Toutefois, vous n'aurez pas l'autorisation d'exécuter des tâches si vous disposez de plus de deux conteneurs avec `log-driver-buffer-limit` défini sur 300 Mio (300 Mio * 2 > 512 Mio).

Utilisation de bibliothèques d'enregistreurs Fluent ou Log4j sur TCP

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.

`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.

- the section called “AWS pour Fluent Bit image”
- the section called “Spécifier une FireLens configuration dans une définition de tâche”
- the section called “Filtrage Fluentd et Fluent Bit journaux ”
- the section called “Exemples de définitions de tâches liées aux options de journalisation”

AWS pour Fluent Bit image

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`x86-64architecture 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'AMD64architecture 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
```

Spécifier une FireLens configuration dans une définition de tâche

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. Pour plus d'informations sur les autorisations nécessaires à un rôle de tâche IAM, veuillez consulter Autorisations IAM requises.

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 plus d'informations, consultez Création d'une définition de tâche 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 utilisant le mode réseau `host`, il s'agit du groupe de sécurité associé à l'instance Amazon EC2 qui héberge 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` : ID de l'instance Amazon EC2 sur laquelle le conteneur est hébergé. Ce champ est uniquement valide pour les tâches qui utilisent le type de lancement EC2.

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": [
        {
            "essential": true,
            "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-
bit:stable",
            "name": "log_router",
            "firelensConfiguration": {
                  "type":"fluentbit",
                   "options":{
                       "enable-ecs-log-metadata":"true"
            },
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "firelens-container",
                    "awslogs-region": "us-west-2",
                    "awslogs-create-group": "true",
                    "awslogs-stream-prefix": "firelens"
                }
            },
            "memoryReservation": 50
        },
        {
            "essential": true,
            "image": "httpd",
            "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. Cette configuration n'est pas spécifiée dans votre définition de tâche.

Spécification d'un fichier de configuration personnalisé

En plus du fichier de configuration généré automatiquement que FireLens crée en votre nom, vous pouvez également 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 Fichier de configuration Fluent Bit.

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. Pour plus d'informations, consultez Autorisations IAM requises.

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:::mybucket/fluent.conf | filepath"
```

```
            }
         }
      }
   ]
}
```

> **Note**
>
> Les tâches hébergées sur AWS Fargate ne prennent en charge que le type `file` de fichier de configuration.

Filtrage Fluentd et Fluent Bit journaux

Il peut arriver que vous souhaitiez filtrer les journaux envoyés, par exemple envoyer des journaux contenant une erreur spécifique.

Fluentdet Fluent Bit les deux prennent en charge le filtrage des journaux en fonction de leur contenu. FireLens fournit une méthode simple pour activer ce filtrage.

Vous pouvez les configurer dans les options LogConfiguration d'une définition de conteneur en spécifiant les expressions régulières correspondantes.

La clé `exclude-pattern` entraîne la suppression de tous les journaux correspondant à son expression régulière. Le `include-pattern` seul envoie des journaux qui correspondent à l'expression régulière spécifiée. Vous pouvez utiliser ces touches séparément ou ensemble

L'exemple suivant illustre l'utilisation de ce filtre.

```
{
   "containerDefinitions":[
      {
         "logConfiguration":{
            "logDriver":"awsfirelens",
            "options":{
               "@type":"cloudwatch_logs",
               "log_group_name":"firelens-testing",
               "auto_create_stream":"true",
               "use_tag_as_stream":"true",
               "region":"us-west-2",
```

```
                "exclude-pattern":"^[a-z][aeiou].*$",
                "include-pattern":"^.*[aeiou]$"
            }
         }
      }
   ]
}
```

Exemples de définitions de tâches liées aux options de journalisation

Voici quelques exemples de définitions de tâches illustrant les options de routage des journaux personnalisées courantes. Pour plus d'exemples, consultez les FireLensexemples d'Amazon ECS sur GitHub.

> **Note**
>
> Le paramètre de définition de `logConfiguration` tâche suivant présenté dans ces exemples est utilisé pour envoyer vos journaux AWS for Fluent Bit à CloudWatch. AWS recommande cette configuration afin que vous disposiez d'informations supplémentaires AWS pour résoudre les problèmes liés CloudWatch à Fluent Bit.
>
> ```
> "logConfiguration": {
>           "logDriver": "awslogs",
>           "options": {
>               "awslogs-group": "firelens-container",
>               "awslogs-region": "us-west-2",
>               "awslogs-create-group": "true",
>               "awslogs-stream-prefix": "firelens"
>           }
>       },
> ```

Rubriques

- Transférer des journaux vers CloudWatch Logs
- Transfert des journaux vers un flux de diffusion Amazon Data Firehose
- Transférer des journaux vers un domaine Amazon OpenSearch Service
- Analyse des journaux de conteneur qui sont JSON sérialisés
- Transfert vers un Fluentd ou Fluent Bit externe

Transférer des journaux vers CloudWatch Logs

> **Note**
>
> Pour plus d'exemples, consultez les FireLens exemples d'Amazon ECS sur GitHub.

L'exemple de définition de tâche suivant montre comment spécifier une configuration de journal qui transfère les journaux à un groupe de CloudWatch journaux de journaux. Pour plus d'informations, consultez Qu'est-ce qu'Amazon CloudWatch Logs ? dans le guide de l'utilisateur d'Amazon CloudWatch Logs.

Dans les options de configuration du journal, spécifiez le nom du groupe de journaux et la Région AWS dans laquelle il existe. Pour que Fluent Bit crée le groupe de journaux en votre nom, spécifiez`"auto_create_group":"true"`, pour définir l' fluentd-buffer-limit utilisation`log-driver-buffer-limit`. Vous pouvez également spécifier l'ID de tâche comme préfixe de flux de journaux, qui facilite le filtrage. Pour plus d'informations, voir Plug-in Fluent Bit pour CloudWatch les journaux.

```
{
    "family": "firelens-example-cloudwatch",
    "taskRoleArn": "arn:aws:iam::123456789012:role/ecs_task_iam_role",
    "containerDefinitions": [
        {
            "essential": true,
            "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-
bit:latest",
            "name": "log_router",
            "firelensConfiguration": {
                "type": "fluentbit"
            },
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "firelens-container",
                    "awslogs-region": "us-west-2",
                    "awslogs-create-group": "true",
                    "awslogs-stream-prefix": "firelens"
                }
            },
            "memoryReservation": 50
```

```
        },
        {
            "essential": true,
            "image": "httpd",
            "name": "app",
            "logConfiguration": {
                "logDriver":"awsfirelens",
                "options": {
                   "Name": "cloudwatch",
                   "region": "us-west-2",
                   "log_group_name": "firelens-blog",
                   "auto_create_group": "true",
                   "log_stream_prefix": "from-fluent-bit",
                   "log-driver-buffer-limit": "2097152"
                }
            },
            "memoryReservation": 100
            }
    ]
}
```

Transfert des journaux vers un flux de diffusion Amazon Data Firehose

> **Note**
>
> Pour plus d'exemples, consultez les FireLens exemples d'Amazon ECS sur GitHub.

L'exemple de définition de tâche suivant montre comment spécifier une configuration de journal qui transfère les journaux vers un flux de diffusion Amazon Data Firehose. Le flux de diffusion Firehose doit déjà exister. Pour plus d'informations, consultez la section Création d'un flux de diffusion Amazon Data Firehose dans le manuel du développeur Amazon Data Firehose.

Dans les options de configuration du journal, spécifiez le nom du flux de remise et la région dans laquelle il existe. Pour plus d'informations, consultez Plug-in Fluent Bit pour Amazon Kinesis Firehose.

```
{
 "family": "firelens-example-firehose",
 "taskRoleArn": "arn:aws:iam::123456789012:role/ecs_task_iam_role",
 "containerDefinitions": [
  {
```

```
    "essential": true,
    "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-bit:stable",
    "name": "log_router",
    "firelensConfiguration": {
     "type": "fluentbit"
    },
    "logConfiguration": {
     "logDriver": "awslogs",
     "options": {
      "awslogs-group": "firelens-container",
      "awslogs-region": "us-west-2",
      "awslogs-create-group": "true",
      "awslogs-stream-prefix": "firelens"
     }
    },
    "memoryReservation": 50
    },
    {
     "essential": true,
     "image": "httpd",
     "name": "app",
     "logConfiguration": {
      "logDriver":"awsfirelens",
      "options": {
      "Name": "firehose",
      "region": "us-west-2",
      "delivery_stream": "my-stream"
     }
    },
    "memoryReservation": 100
            }
 ]
}
```

Transférer des journaux vers un domaine Amazon OpenSearch Service

> **Note**
>
> Pour plus d'exemples, consultez les FireLens exemples d'Amazon ECS sur GitHub.

L'exemple de définition de tâche suivant montre comment spécifier une configuration de journal qui transfère les journaux vers un domaine Amazon OpenSearch Service ;. Le domaine Amazon

OpenSearch Service doit déjà exister. Pour plus d'informations, consultez la section Qu'est-ce qu'Amazon OpenSearch Service dans le manuel Amazon OpenSearch Service Developer Guide.

Dans les options de configuration du journal, spécifiez les options de journal requises pour l'intégration OpenSearch du service. Pour plus d'informations, consultez Fluent Bit pour Amazon OpenSearch Service.

```
{
    "family": "firelens-example-opensearch",
    "taskRoleArn": "arn:aws:iam::123456789012:role/ecs_task_iam_role",
    "containerDefinitions": [
        {
            "essential": true,
            "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-
bit:stable",
            "name": "log_router",
            "firelensConfiguration": {
                "type": "fluentbit"
            },
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "firelens-container",
                    "awslogs-region": "us-west-2",
                    "awslogs-create-group": "true",
                    "awslogs-stream-prefix": "firelens"
                }
            },
            "memoryReservation": 50
        },
        {
            "essential": true,
            "image": "httpd",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "es",
                    "Host": "vpc-fake-domain-ke7thhzo07jawrhmz6mb7ite7y.us-
west-2.es.amazonaws.com",
                    "Port": "443",
                    "Index": "my_index",
                    "Type": "my_type",
```

```
                    "AWS_Auth": "On",
                    "AWS_Region": "us-west-2",
                    "tls": "On"
                }
            },
            "memoryReservation": 100
        }
    ]
}
```

Analyse des journaux de conteneur qui sont JSON sérialisés

> **Note**
>
> Pour plus d'exemples, consultez les FireLens exemples d'Amazon ECS sur GitHub.

À partir de AWS la version 1.3 de Fluent Bit, un analyseur JSON est inclus dans l'image AWS de Fluent Bit. L'exemple suivant montre comment référencer l'analyseur JSON dans la FireLens configuration de votre définition de tâche.

```
"firelensConfiguration": {
    "type": "fluentbit",
    "options": {
        "config-file-type": "file",
        "config-file-value": "/fluent-bit/configs/parse-json.conf"
    }
},
```

Le fichier de configuration Fluent Bit analyse tous les journaux au format JSON (par exemple, si les journaux de votre destination ressemblaient à ce qui suit sans analyse JSON).

```
{
    "source": "stdout",
    "log": "{\"requestID\": \"b5d716fca19a4252ad90e7b8ec7cc8d2\", \"requestInfo\":
 {\"ipAddress\": \"204.16.5.19\", \"path\": \"/activate\", \"user\": \"TheDoctor\"}}",
    "container_id": "e54cccfac2b87417f71877907f6787906842004282806 7ae0867e60a63529d35",
    "container_name": "/ecs-demo-6-container2-a4eafbb3d4c7f1e16e00"
    "ecs_cluster": "mycluster",
    "ecs_task_arn": "arn:aws:ecs:us-east-2:01234567891011:task/
mycluster/3de392df-6bfa-470b-97ed-aa6f482cd7a6",
    "ecs_task_definition": "demo:7"
```

```
    "ec2_instance_id": "i-06bc83dbc2ac2fdf8"
}
```

Avec l'analyse JSON, le journal se présente comme suit.

```
{
    "source": "stdout",
    "container_id": "e54cccfac2b87417f71877907f6787906842004282806 7ae0867e60a63529d35",
    "container_name": "/ecs-demo-6-container2-a4eafbb3d4c7f1e16e00"
    "ecs_cluster": "mycluster",
    "ecs_task_arn": "arn:aws:ecs:us-east-2:01234567891011:task/
mycluster/3de392df-6bfa-470b-97ed-aa6f482cd7a6",
    "ecs_task_definition": "demo:7"
    "ec2_instance_id": "i-06bc83dbc2ac2fdf8"
    "requestID": "b5d716fca19a4252ad90e7b8ec7cc8d2",
    "requestInfo": {
        "ipAddress": "204.16.5.19",
        "path": "/activate",
        "user": "TheDoctor"
    }
}
```

Le JSON sérialisé est développé en champs de niveau supérieur dans la sortie JSON finale. Pour plus d'informations sur l'analyse JSON, veuillez consulter la rubrique Analyseur dans la documentation Fluent Bit.

Transfert vers un Fluentd ou Fluent Bit externe

> **Note**
>
> Pour plus d'exemples, consultez les FireLens exemples d'Amazon ECS sur GitHub.

L'exemple de définition de tâche suivant montre comment spécifier une configuration de journal qui transfère les journaux vers un hôte Fluentd ou Fluent Bit externe. Spécifiez `host` et `port` pour votre environnement.

```
{
 "family": "firelens-example-forward",
 "taskRoleArn": "arn:aws:iam::123456789012:role/ecs_task_iam_role",
 "containerDefinitions": [
```

```
  {
   "essential": true,
   "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-bit:stable",
   "name": "log_router",
   "firelensConfiguration": {
    "type": "fluentbit"
   },
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-group": "firelens-container",
     "awslogs-region": "us-west-2",
     "awslogs-create-group": "true",
     "awslogs-stream-prefix": "firelens"
    }
   },
   "memoryReservation": 50
   },
   {
    "essential": true,
    "image": "httpd",
    "name": "app",
    "logConfiguration": {
     "logDriver":"awsfirelens",
     "options": {
     "Name": "forward",
     "Host": "fluentdhost",
     "Port": "24224"
    }
   },
   "memoryReservation": 100
  }
 ]
}
```

Authentification de registre privé pour les tâches

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 fonction est prise en charge par les tâches hébergées sur Fargate, les instances Amazon EC2 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 des instances Amazon EC2, cette fonction exige nécessite la version `1.19.0` 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 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 plateforme Linux Fargate.

Dans votre définition du conteneur, spécifiez l'objet `repositoryCredentials` avec les détails du secret que vous avez créé. Le secret auquel vous faites référence 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 *`private-repo`* par le nom d'hôte du dépôt privé et *`private-image par le nom de l'image`*.

```
"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 est prise en charge uniquement pour les tâches hébergées sur des instances Amazon EC2. Pour plus d'informations, consultez Authentification de registre privé pour les instances de conteneur.

Pour utiliser un registre privé

1. Le rôle d'exécution de tâche Amazon ECS est requis pour utiliser cette fonction. Cela permet à l'agent de conteneur d'extraire l'image de conteneur. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

   Pour fournir l'accès aux secrets que vous créez, ajoutez les autorisations suivantes en tant que politique en ligne au rôle d'exécution de tâche. Pour plus d'informations, consultez Ajout et suppression de politiques IAM.

   - `secretsmanager:GetSecretValue`
   - `kms:Decrypt` : requis uniquement si votre clé 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 plus d'informations, consultez the section called “Création d'une définition de tâche à l'aide de la console”.

Utiliser les paramètres de définition des tâches pour transmettre des variables d'environnement à un conteneur

> **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 plus d'informations, consultez Transmission de données sensibles à un conteneur.
> 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 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"
                }
            ],
            ...
        }
    ],
    ...
}
```

Utiliser un fichier pour transmettre des variables d'environnement à un conteneur

> **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 plus d'informations, consultez Transmission de données sensibles à un conteneur.

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 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.

Considérations

Tenez compte des éléments suivants lors de la spécification d'un fichier de variable d'environnement dans une définition de conteneur.

- Pour toute tâche Amazon ECS hébergé sur Amazon EC2, vos instances de conteneur nécessitent une version `1.39.0` ou ultérieure de l'agent de conteneur pour utiliser cette fonction. 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 plus d'informations, consultez Versions de plateforme Linux Fargate.

  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.
- 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 de plus amples informations sur la syntaxe du fichier de variable d'environnement, consultez Déclarer les variables d'environnement par défaut dans le fichier.

  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 comme un fichier d'environnement Docker natif.
  - La gestion de l'échappement dans shell n'est pas prise en charge.
  - Le point d'entrée du conteneur interprète les valeurs `VARIABLE`.

Autorisations IAM requises

Le rôle d'exécution de tâche Amazon ECS est requis pour utiliser cette fonction. Cela permet à l'agent de conteneur d'extraire le fichier de variable d'environnement à partir d'Amazon S3. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Pour fournir l'accès aux objets Amazon S3 que vous créez, ajoutez manuellement les autorisations suivantes en tant que stratégie en ligne au rôle d'exécution de tâche. Utilisez le paramètre `Resource` pour étendre l'autorisation aux compartiments Amazon S3 qui contiennent les fichiers de variables d'environnement. Pour plus d'informations, consultez Ajout et suppression de politiques IAM.

- `s3:GetObject`
- `s3:GetBucketLocation`

Dans l'exemple suivant, les autorisations sont ajoutées à la stratégie en ligne.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket/folder_name/env_file_name"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::examplebucket"
      ]
    }
  ]
}
```

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:::s3_bucket_name/envfile_object_name.env",
                    "type": "s3"
                }
            ],
            ...
        }
    ],
    ...
}
```

Transmission de données sensibles à un conteneur

Vous pouvez transmettre en toute sécurité des données sensibles, telles que des informations d'identification vers une base de données, dans votre conteneur.

Vous pouvez utiliser AWS Secrets Manager ou en tant que paramètre dans AWS Systems Manager Parameter Store pour stocker le secret.

Vous pouvez récupérer les secrets par programmation depuis l'application ou à l'aide de variables d'environnement.

Pour commencer, stockez d'abord les données sensibles en tant que secret AWS Secrets Manager ou en tant que paramètre dans AWS Systems Manager Parameter Store. Utilisez ensuite l'une des méthodes suivantes pour exposer le secret dans le conteneur.

Rubriques

- Récupérez les secrets de Secrets Manager par programmation
- Récupérez les AWS Systems Manager paramètres du Parameter Store par programmation
- Récupérez les secrets de Secrets Manager via des variables d'environnement
- Récupération de AWS Systems Manager paramètres via des variables d'environnement
- Récupérez les secrets pour la configuration de la journalisation

Récupérez les secrets de Secrets Manager par programmation

Utilisez Secrets Manager pour protéger les données sensibles et effectuer la rotation, la gestion et la récupération des identifiants de base de données, des clés API et d'autres secrets tout au long de leur cycle de vie.

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.

Considérations

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.

Autorisations IAM requises

Pour utiliser cette fonctionnalité, vous devez disposer du rôle de tâche Amazon ECS et le référencer dans votre définition de tâche. Pour plus d'informations, consultez Rôle IAM de tâche.

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"
      ]
    }
  ]
}
```

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 les API de Secrets Manager directement depuis votre application. Pour plus d'informations sur la manière de mettre à jour votre code pour demander le secret, consultez la section Récupération des secrets à partir de AWS Secrets Manager dans le Guide de l'utilisateur AWS Secrets Manager .

Pour récupérer les données sensibles stockées dans le AWS Secrets Manager, consultez les exemples de code pour AWS Secrets Manager l'utilisation AWS des SDK dans la bibliothèque de codes d'exemples de code AWS SDK.

Récupérez les AWS Systems Manager paramètres du Parameter Store par programmation

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 identifiants d'instance Amazon Elastic Compute Cloud (Amazon EC2) et les ID d'AMI, ainsi que les codes de licence sous forme de valeurs de paramètres. Vous pouvez stocker ces valeurs sous forme de texte brut ou de données chiffrées.

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 paramètre Systems Manager Parameter Store est ultérieurement mis à jour, l'application récupère automatiquement la dernière version.

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.

Considérations

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.

Autorisations IAM requises

Pour utiliser cette fonctionnalité, vous devez disposer du rôle de tâche Amazon ECS et le référencer dans votre définition de tâche. Cela permet à l'agent de conteneur d'extraire les ressources Systems Manager nécessaires. Pour plus d'informations, consultez Rôle IAM de tâche.

> **Important**
>
> Pour les tâches qui utilisent le type de lancement EC2, vous devez utiliser la variable de configuration de l'agent ECS `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` pour utiliser cette fonction. 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 plus d'informations, consultez 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"
      ]
    }
  ]
}
```

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 pour Systems Manager utilisant AWS des SDK dans la bibliothèque de codes AWS SDK Code Examples.

Récupérez les secrets de Secrets Manager via des variables d'environnement

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 ou une version spécifique d'un secret à injecter. Cela vous aide à contrôler les données sensibles exposées à votre conteneur. Pour de plus amples informations sur la gestion des versions de secrets, veuillez consulter Termes et concepts clés pour AWS Secrets Manager dans le Guide de l'utilisateur AWS Secrets Manager .

Considérations

Les informations suivantes doivent être prises 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.
- 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 plateforme Linux Fargate.
  - 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 plateforme Linux Fargate.
- Pour les tâches Amazon ECS sur EC2, les informations suivantes doivent être prises 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 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 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. Cela peut être fait avec les données utilisateur en utilisant la syntaxe suivante :

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster <cluster name> -EnableTaskIAMRole -LoggingDrivers
 '["json-file","awslogs"]'
</powershell>
```

Autorisations IAM

Pour utiliser cette fonction, vous devez disposer du rôle d'exécution de tâche Amazon ECS et le référencer dans votre définition de tâche. Pour plus d’informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Pour fournir l'accès aux secrets Secrets Manager que vous créez, ajoutez manuellement les autorisations suivantes en tant que stratégie en ligne au rôle d'exécution de tâche. Pour plus d'informations, consultez Ajout et suppression de politiques IAM.

- `secretsmanager:GetSecretValue` : obligatoire si vous faites référence à un secret Secrets Manager. 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 managée client 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": [
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name",
        "arn:aws:kms:region:aws_account_id:key/key_id"
      ]
    }
  ]
}
```

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. 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 .

`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.

Les ID de version sont utilisés 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 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 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"
    }]
```

```
  }]
}
```

Créer une définition de tâche faisant référence à des données sensibles

Vous pouvez utiliser la console Amazon ECS pour créer une définition de tâche qui fait référence à un secret Secrets Manager.

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 (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 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 uniquement à l'aide du nom de registre Amazon ECR Public. Par exemple, si `public.ecr.aws/ecs/amazon-ecs-agent:latest` est spécifié, le conteneur Amazon Linux hébergé sur Amazon ECR Public Gallery est utilisé. Pour tous les autres référentiels, spécifiez le référentiel en utilisant les formats `repository-url/image:tag` ou `repository-url/image@digest`.

   c. Pour Essential container (Conteneur essentiel), si deux conteneurs ou plus sont définis dans votre définition de tâche, vous pouvez spécifier si le conteneur doit être considéré comme essentiel. Lorsqu'un conteneur est marqué comme essentiel, s'il s'arrête, alors la tâche s'arrête aussi. Chaque définition de tâche doit contenir au moins un conteneur essentiel.

   d. 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. Vous sélectionnez le mode réseau bridge (pont) sur la page suivante. Une

fois que vous l'avez sélectionné, choisissez Previous (Précédent), puis pour Host port (Port hôte), spécifiez le numéro de port sur l'instance de conteneur à réserver pour votre conteneur.

Choisissez Add more port mappings (Ajouter d'autres mappages de ports) pour spécifier des mappages de ports de conteneur supplémentaires.

e. Pour que des données sensibles soient injectées en tant que variables d'environnement, sous Environment (Environnement), pour Environment variables (Variables d'environnement), complétez les champs suivants :

   i. Pour Key (Clé), saisissez le nom de la variable d'environnement à définir dans le conteneur. Cela correspond au champ `name` de la section `secrets` d'une définition de conteneur.

   ii. Pour Type de valeur, sélectionnez ValueFrom. Pour Add value (Ajouter une valeur), saisissez l'ARN du secret Secrets Manager qui contient les données à présenter à votre conteneur en tant que variable d'environnement.

f. (Facultatif) Choisissez Add more containers (Ajouter des conteneurs supplémentaires) pour ajouter des conteneurs supplémentaires à la définition de tâche. Choisissez Next (Suivant) une fois que tous les conteneurs ont été définis.

6. Pour l'environnement de l'application, la taille de la tâche et la taille du conteneur, renseignez les champs obligatoires restants et tous les champs facultatifs.

7. (Facultatif) Développez la section Task roles, network mode (Rôles de tâches, mode réseau) pour spécifier les éléments suivants :

   • Pour Task role (Rôle de tâche), choisissez le rôle IAM à assigner à la tâche. Un rôle IAM de tâche fournit des autorisations pour les conteneurs d'une tâche à appeler les API d' AWS .

8. (Facultatif) La section Storage (Stockage) permet d'étendre la quantité de magasins éphémères pour les tâches hébergées sur Fargate et d'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 GiB pour vos tâches Fargate pour Amount (Quantité), spécifiez une valeur allant jusqu'à 200 GiB.

9. Pour les données sensibles référencées dans la configuration de journal pour un conteneur, sous Use log collection (Utiliser la collecte de journaux), pour Log configuration (Configuration de journal), complétez la configuration suivante :

a. Sélectionnez l'option journal, puis sous Clé, choisissez Ajouter.

b. Pour Key (Clé), saisissez le nom de l'option de configuration de journaux à définir.

c. Pour Value (Valeur), saisissez l'ARN complet du secret Secrets Manager qui contient les données à présenter à votre configuration de journal en tant qu'option de journal.

d. Pour Type de valeur, sélectionnez ValueFrom.

10. Choisissez Next (Suivant) pour passer en revue la définition de tâche.
11. Sur la page Review and create (Vérifier et créer), passez en revue chaque section de définition de tâche. Vous pouvez choisir Edit (Modifier) pour effectuer des changements. Une fois la définition de tâche terminée, choisissez Create (Créer) pour enregistrer la définition de tâche.

Récupération de AWS Systems Manager paramètres via des variables d'environnement

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.

Considérations

Les points suivants doivent être pris en compte lors de l'utilisation d'une variable d'environnement pour injecter un AWS Systems Manager secret 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 plateforme Linux Fargate.
  - 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 plateforme Linux Fargate.

- Pour les tâches Amazon ECS sur EC2, les informations suivantes doivent être prises 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. AWS Systems 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 Systems Manager .
- 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. Cela peut être fait avec les données utilisateur en utilisant la syntaxe suivante :

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster <cluster name> -EnableTaskIAMRole -LoggingDrivers
 '["json-file","awslogs"]'
</powershell>
```

Autorisations IAM

Pour utiliser cette fonction, vous devez disposer du rôle d'exécution de tâche Amazon ECS et le référencer dans votre définition de tâche. Cela permet à l'agent du conteneur de récupérer les AWS Systems Manager ressources nécessaires. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

> **Important**
>
> Pour les tâches qui utilisent le type de lancement EC2, vous devez utiliser la variable de configuration de l'agent ECS `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true`

pour utiliser cette fonction. 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 plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

Pour donner accès aux AWS Systems Manager paramètres du magasin de paramètres que vous créez, ajoutez manuellement les autorisations suivantes au rôle d'exécution des tâches. 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"
      ]
```

```
    }
  ]
}
```

Créer le paramètre AWS 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 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 plus d'informations, consultez `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"
    }]
  }]
}
```

Créer une définition de tâche faisant référence à des données sensibles

Vous pouvez utiliser la console Amazon ECS pour créer une définition de tâche qui fait référence à un paramètre Systems Manager Parameter Store.

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 (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 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 uniquement à l'aide du nom de registre Amazon ECR Public. Par exemple, si `public.ecr.aws/ecs/amazon-ecs-agent:latest` est spécifié, le conteneur Amazon Linux hébergé sur Amazon ECR Public Gallery est utilisé. Pour tous les autres référentiels, spécifiez le référentiel en utilisant les formats `repository-url/image:tag` ou `repository-url/image@digest`.

   c. Pour Essential container (Conteneur essentiel), si deux conteneurs ou plus sont définis dans votre définition de tâche, vous pouvez spécifier si le conteneur doit être considéré comme essentiel. Lorsqu'un conteneur est marqué comme essentiel, s'il s'arrête, alors la tâche s'arrête aussi. Chaque définition de tâche doit contenir au moins un conteneur essentiel.

   d. 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. Vous sélectionnez le mode réseau bridge (pont) sur la page suivante. Une fois que vous l'avez sélectionné, choisissez Previous (Précédent), puis pour Host port (Port hôte), spécifiez le numéro de port sur l'instance de conteneur à réserver pour votre conteneur.

      Choisissez Add more port mappings (Ajouter d'autres mappages de ports) pour spécifier des mappages de ports de conteneur supplémentaires.

   e. Pour que des données sensibles soient injectées en tant que variables d'environnement, sous Environment (Environnement), pour Environment variables (Variables d'environnement), complétez les champs suivants :

i. Pour Key (Clé), saisissez le nom de la variable d'environnement à définir dans le conteneur. Cela correspond au champ `name` de la section `secrets` d'une définition de conteneur.

ii. Pour Type de valeur, sélectionnez ValueFrom. Pour Value, entrez le nom ou l'ARN complet du AWS Systems Manager paramètre Parameter Store qui contient les données à présenter à votre configuration de journal sous forme d'option de journal.

> **Note**
>
> 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 secret. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

f. (Facultatif) Choisissez Add more containers (Ajouter des conteneurs supplémentaires) pour ajouter des conteneurs supplémentaires à la définition de tâche. Choisissez Next (Suivant) une fois que tous les conteneurs ont été définis.

6. Pour l'environnement de l'application, la taille de la tâche et la taille du conteneur, renseignez les champs obligatoires restants et tous les champs facultatifs.

7. (Facultatif) Développez la section Task roles, network mode (Rôles de tâches, mode réseau) pour spécifier les éléments suivants :

   - Pour Task role (Rôle de tâche), choisissez le rôle IAM à assigner à la tâche. Un rôle IAM de tâche fournit des autorisations pour les conteneurs d'une tâche à appeler les API d' AWS .

8. (Facultatif) La section Storage (Stockage) permet d'étendre la quantité de magasins éphémères pour les tâches hébergées sur Fargate et d'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 GiB pour vos tâches Fargate pour Amount (Quantité), spécifiez une valeur allant jusqu'à 200 GiB.

9. Pour les données sensibles référencées dans la configuration de journal pour un conteneur, sous Use log collection (Utiliser la collecte de journaux), pour Log configuration (Configuration de journal), complétez la configuration suivante :

   a. Sélectionnez l'option journal, puis sous Clé, choisissez Ajouter.

b. Pour Key (Clé), saisissez le nom de l'option de configuration de journaux à définir.

c. Pour Value, entrez le nom ou l'ARN complet du AWS Systems Manager paramètre Parameter Store qui contient les données à présenter à votre configuration de journal sous forme d'option de journal.

> **Note**
>
> 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 secret. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

d. Pour Type de valeur, sélectionnez ValueFrom.

10. Choisissez Next (Suivant) pour passer en revue la définition de tâche.
11. Sur la page Review and create (Vérifier et créer), passez en revue chaque section de définition de tâche. Vous pouvez choisir Edit (Modifier) pour effectuer des changements. Une fois la définition de tâche terminée, choisissez Create (Créer) pour enregistrer la définition de tâche.

Récupérez les secrets pour la configuration de la journalisation

Utilisation de Secrets Manager

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.

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"
      }]
    }]
  }]
}
```

En utilisant AWS 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"
      }]
    }]
  }]
}
```

Exemples de définitions de tâches

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 à l'aide de la console et Paramètres de définition de tâche.

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
- 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": "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"]
}
```

splunkpilote 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 plus d'informations, consultez Transmission de données sensibles à un conteneur.

```
"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 plus d'informations, consultez Transmission de données sensibles à un conteneur.

```
"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 plus d'informations, consultez `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 plus d'informations, consultez Création d'une définition de tâche 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 plus d'informations, consultez Rôle IAM de tâche.

```
{
    "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 `google.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 plus d'informations, consultez 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"
}
```

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 et capacité 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 :
  - Instances Amazon EC2 dans le cloud AWS
  - Sans serveur (AWS Fargate (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 instances Amazon EC2 pour la capacité, le sous-réseau peut se trouver dans des zones de disponibilité, des zones locales, des zones Wavelength 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.

Concepts des fournisseurs de capacité

Les fournisseurs de capacité sont constitués des composants suivants.

Fournisseur de capacité

Un fournisseur de capacité définit la capacité du cluster qu'Amazon ECS augmente et réduit par rapport à l'infrastructure que vous spécifiez. Vous devez d'abord associer le fournisseur de capacité à un cluster avant de l'utiliser.

Vous utilisez un fournisseur de capacité dans une stratégie de fournisseur de capacité pour déterminer l'infrastructure sur laquelle une tâche s'exécute. Chaque tâche doit avoir une stratégie de fournisseur de capacité, un type de lancement ou utiliser la stratégie de fournisseur de capacité par défaut associée au cluster sélectionné. Vous devez faire référence à la stratégie du fournisseur de capacité et non au fournisseur de capacité. Si une tâche utilise un type de

lancement, la capacité qu'elle utilise n'est prise en compte par aucun fournisseur de capacité du cluster.

Car AWS Fargate les fournisseurs de capacités sont un `FARGATE` fournisseur `FARGATE_SPOT` de capacité qui AWS crée. Vous associez le fournisseur de capacité à votre cluster, puis vous l'ajoutez à une stratégie de fournisseur de capacité.

Pour les utilisateurs d'Amazon ECS sur Amazon EC2, un fournisseur de capacité se compose d'un nom et d'un groupe Auto Scaling. Un fournisseur de capacité comprend également tous les paramètres de mise à l'échelle gérée et de protection contre la résiliation gérée. Lorsque vous activez la mise à l'échelle gérée, Amazon ECS augmente ou réduit les groupes Auto Scaling à votre place.

Stratégie de fournisseur de capacité par défaut

Vous pouvez associer une stratégie de fournisseur de capacité par défaut à un cluster Amazon ECS. Une fois cela fait, Amazon ECS utilise une stratégie de fournisseur de capacité par défaut lors de la création d'un service ou l'exécution d'une tâche autonome dans le cluster et lorsqu'aucun fournisseur de capacité personnalisé ou type de lancement n'est spécifié. Nous vous recommandons de définir une stratégie de fournisseur de capacité par défaut pour chaque cluster.

Stratégie de fournisseur de capacité

Une stratégie de fournisseur de capacité consiste en un ou plusieurs fournisseurs de capacité. Vous pouvez spécifier des valeurs de base et de poids facultatives pour un contrôle plus précis. Une stratégie de fournisseur de capacité fait partie de la configuration d'un cluster, d'un service ou d'une tâche. Toutefois, vous ne pouvez pas créer des stratégies de fournisseur de capacité réutilisables. La stratégie de fournisseur de capacité de chaque cluster, service ou tâche est indépendante.

Si la stratégie de fournisseur de capacité par défaut d'un cluster ne répond à vos besoins de capacité, spécifiez une stratégie de fournisseur de capacité personnalisée lors de la création d'un service ou de l'exécution d'une tâche autonome.

> **Important**
>
> Lorsque vous définissez un type de lancement plutôt qu'une stratégie de fournisseur de capacité pour des tâches dans des clusters où la capacité est gérée par des fournisseurs de capacité, ces tâches ne sont pas comptées parmi les actions de mise à l'échelle du fournisseur de capacité.

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.

Fournisseurs de capacité Amazon ECS

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.

Des fournisseurs de capacité sont disponibles pour les tâches qui s'exécutent sur Fargate ou sur des instances Amazon EC2. Vous ne pouvez pas utiliser de fournisseurs de capacité pour les tâches exécutées sur des instances de conteneur externes (Amazon ECS Anywhere).

Types de fournisseurs de capacité

Pour les charges de travail Amazon ECS hébergées sur Fargate, les fournisseurs de capacité prédéfinis suivants sont disponibles :

- Fargate
- Fargate Spot

Pour les charges de travail Amazon ECS hébergées sur des instances Amazon EC2, vous devez créer et gérer un fournisseur de capacité qui comprend les composants suivants :

- Un nom
- Un groupe Auto Scaling
- Les paramètres de mise à l'échelle gérée et de protection contre la résiliation gérée.

Vous pouvez créer le groupe Auto Scaling lorsque vous créez le cluster, ou vous pouvez créer le groupe avant le cluster, puis spécifier le nom du groupe lorsque vous créez le cluster.

Considérations relatives aux fournisseurs de capacité

Prenez les points suivants en compte lors de l'utilisation de fournisseurs de capacité :

- Un fournisseur de capacité doit être associé à un cluster avant d'être spécifié dans une stratégie de fournisseur de capacité.
- Lorsque vous spécifiez une stratégie de fournisseur de capacité, le nombre de fournisseurs de capacité que vous pouvez spécifier est limité à 20.
- 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. De plus, les fournisseurs de capacité dont le poids est nul ne sont pas utilisés pour placer 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.
- Lorsque vous utilisez la protection contre la résiliation gérée, vous devez également utiliser la mise à l'échelle gérée. Sinon, la protection contre la résiliation gérée ne fonctionne pas.

Rubriques

- AWS Fargate fournisseurs de capacité
- Fournisseurs de capacité de groupe Amazon EC2 Auto Scaling

AWS Fargate fournisseurs de capacité

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 AWS a de nouveau besoin de la capacité, vos tâches sont interrompues avec un avertissement de deux minutes.

Considérations relatives aux fournisseurs de capacité Fargate

Prenez les points suivants en compte lors de l'utilisation de fournisseurs de capacité Fargate :

- Les conteneurs Windows sur Fargate ne prennent pas en charge le fournisseur de capacité Fargate Spot.
- Les tâches Linux avec l'architecture ARM64 ne prennent pas en charge le fournisseur de capacité Fargate Spot. Fargate Spot prend uniquement en charge les tâches Linux avec l'architecture X86_64.
- Vous ne devez pas créer les fournisseurs de capacité Fargate et Fargate Spot. Ils sont disponibles pour tous les comptes. Pour les utiliser, il vous suffit de les associer à un cluster.
- Pour associer les fournisseurs de capacité Fargate et Fargate Spot à un cluster, vous devez utiliser l'API Amazon ECS ou. AWS CLI Vous ne pouvez pas les associer à l'aide de la console.

- Les fournisseurs de capacité Fargate et Fargate Spot sont réservés et ne peuvent pas être supprimés. Vous pouvez toutefois les dissocier d'un cluster à l'aide de l'opération d'API `PutClusterCapacityProviders`.
- Vous pouvez associer un fournisseur de capacité à un cluster existant à l'aide de l'opération d'API `PutClusterCapacityProviders`.
- Si vous utilisez Fargate Spot, votre tâche doit obligatoirement utiliser la version 1.3.0 de la plateforme ou une version ultérieure (Linux). Pour plus d'informations, consultez Versions de plateforme Linux Fargate.
- 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 plus d'informations, consultez 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 de plus amples informations, veuillez consulter Fournisseurs de capacité de groupe Auto Scaling dans le Guide du développeur Amazon Elastic Container Service.

Gestion des notifications de résiliation de Fargate Spot

Ayez conscience des conséquences suivantes, car la capacité Spot peut ne pas être disponible en permanence.

- 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 plus d'informations, consultez 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, consultez la section Création d'une EventBridge règle 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"
        ]
    }
}
```

Fournisseurs de capacité de groupe Amazon EC2 Auto Scaling

Lorsque vous utilisez les instances Amazon EC2 pour votre capacité, vous utilisez les groupes Auto Scaling pour gérer les instances Amazon EC2 enregistrées sur leurs clusters. Auto Scaling permet de vous assurer que vous disposez du bon nombre d'instances Amazon EC2 disponibles pour gérer la charge de l'application.

Vous pouvez utiliser la fonctionnalité de mise à l'échelle gérée pour qu'Amazon ECS gère les actions de mise à l'échelle horizontale ou de montée en puissance du groupe Auto Scaling (mise à l'échelle gérée) ou vous pouvez gérer les actions de mise à l'échelle vous-même. Pour plus d’informations, consultez Auto Scaling de cluster Amazon ECS.

Considérations relatives aux fournisseurs de capacité de groupe Auto Scaling

Les points suivants doivent être pris en compte lors de l'utilisation de fournisseurs de capacité de groupe Auto Scaling dans la console :

- Nous vous recommandons de créer un groupe Auto Scaling à utiliser avec un fournisseur de capacité plutôt que d'en utiliser un déjà existant. Si vous utilisez un groupe Auto Scaling existant, il se peut que les instances Amazon EC2 associées au groupe qui étaient déjà en cours d'exécution et enregistrées sur un cluster Amazon ECS avant le groupe Auto Scaling utilisé pour créer un fournisseur de capacité ne soient pas bien 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 Create Cluster avec l'option Amazon EC2 instances sous Infrastructure, Amazon ECS crée une configuration de lancement Amazon EC2 Auto Scaling et un groupe Auto Scaling en votre nom dans le cadre AWS CloudFormation de la pile. Ils sont précédés du préfixe `EC2ContainerService-<ClusterName>`, ce qui facilite leur identification. Vous pouvez utiliser le groupe Auto Scaling comme fournisseur de capacité pour ce cluster.

- 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. La pondération d'instance n'est pas prise en charge lorsqu'elle est utilisée avec un fournisseur de capacité Amazon ECS.
- 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.
- Lorsque vous utilisez la protection contre la résiliation gérée, vous devez également utiliser la mise à l'échelle gérée. Sinon, la protection contre la résiliation gérée ne fonctionnera pas.
- Lorsque la mise à l'échelle gérée est activée, le fournisseur de capacité du groupe Auto Scaling crée une ressource de politique de mise à l'échelle pour gérer la mise à l'échelle de votre groupe Auto Scaling. Vous pouvez identifier ces ressources à l'aide du préfixe `ECSManaged`.

  Lorsque vous utilisez la protection contre la résiliation gérée, Amazon ECS résilie uniquement les instances EC2 qui n'ont pas de tâches Amazon ECS en cours d'exécution.
- Si la protection contre la résiliation gérée est activée pendant que vous créez un fournisseur de capacité, la protection contre la mise à l'échelle horizontale en charge d'instance doit être activée

pour le groupe Auto Scaling et chaque instance Amazon EC2 du groupe Auto Scaling. Pour plus d'informations, consultez Protection des instances dans le Guide de l'utilisateur AWS Auto Scaling .

- Vous pouvez ajouter un groupe d'instances pré-initialisées à votre groupe Auto Scaling. Un groupe d'instances pré-initialisées est un groupe d'instances Amazon EC2 pré-initialisées qui sont prêtes à être incluses dans le cluster chaque fois que votre application doit monter en puissance. Pour de plus amples informations sur les groupes d'instances pré-initialisées, veuillez consulter Utilisation d'un groupe d'instances pré-initialisées pour votre groupe Auto Scaling.
- 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.
- Le drainage géré des instances est activé par défaut lorsque vous créez un fournisseur de capacité. Nous vous recommandons d'utiliser cette fonctionnalité pour permettre la résiliation progressive des instances Amazon EC2 sans perturber vos charges de travail.

Pour plus d'informations sur la création de modèles de lancement pour Amazon EC2 Auto Scaling, consultez la section Modèles de lancement dans le Guide de l'utilisateur Amazon EC2 Auto Scaling. Pour plus d'informations sur la création de groupe Amazon EC2 Auto Scaling, consultez la section Création de groupes Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Utilisation d'un groupe d'instances pré-initialisées pour votre groupe Auto Scaling

Amazon ECS prend en charge les groupes d'instances pré-initialisées Amazon EC2 Auto Scaling. Un groupe d'instances pré-initialisées est un groupe d'instances Amazon EC2 pré-initialisées prêtes à être mises en service. Chaque fois que votre application doit monter en puissance, Amazon EC2 Auto Scaling utilise les instances pré-initialisées du groupe d'instances pré-initialisées plutôt que de lancer des instances froides, permet l'exécution de tout processus d'initialisation final, puis met l'instance en service.

Pour en savoir plus sur les groupes instances pré-initialisées et comment les ajouter à votre groupe Auto Scaling, consultez la section Groupes d'instances pré-initialisées pour Amazon EC2 Auto Scaling dans le Guide de l'utilisateur 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 groupes d'instances pré-initialisées avec votre cluster Amazon ECS, définissez la variable de configuration de l'agent `ECS_WARM_POOLS_CHECK` sur `true` dans le champ User data (Données utilisateur) du modèle de lancement de votre groupe Amazon EC2 Auto Scaling. Voici un exemple de la façon dont la variable de configuration de l'agent peut être spécifiée dans le champ User data (Données utilisateur) d'un modèle de lancement Amazon EC2.

```
#!/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.

Auto Scaling de cluster Amazon ECS

Amazon ECS peut gérer la mise à l'échelle des instances Amazon EC2 enregistrées sur votre cluster. C'est ce qu'on appelle l'autoscaling du cluster Amazon ECS. Pour ce faire, utilisez un fournisseur de capacité de groupe Auto Scaling Amazon ECS avec la mise à l'échelle gérée activée. Lorsque vous utilisez un fournisseur de capacité de groupe Auto Scaling avec la mise à l'échelle gérée activée, vous définissez un pourcentage cible (la `targetCapacity`) pour l'utilisation des instances de ce groupe Auto Scaling. Amazon ECS crée deux CloudWatch métriques personnalisées et une politique de dimensionnement du suivi des cibles qui s'attache à votre groupe Auto Scaling. Amazon ECS gère ensuite les actions de mise à l'échelle horizontale et de montée en puissance du groupe Auto Scaling en fonction des ressources que vos tâches utilisent auprès de ce fournisseur de capacité. Pour plus d'informations sur les fournisseurs de capacité de groupe Auto Scaling, consultez la section the section called “Fournisseurs de capacité de groupe Amazon EC2 Auto Scaling”.

> **Note**
>
> La mise à l'échelle automatique du cluster Amazon ECS n'est prise en charge qu'avec les fournisseurs de capacité de groupe Auto Scaling. Pour les charges de travail Amazon ECS hébergées sur AWS Fargate, consultezthe section called “AWS Fargate fournisseurs de capacité”.

Voici le flux de travail pour utiliser l'autoscaling de cluster Amazon ECS. Pour plus d'informations, consultez the section called "Activation de l'autoscaling de cluster".

1. Créez un groupe Auto Scaling.
2. Créez un fournisseur de capacité qui utilise ce groupe Auto Scaling.
3. Activez la mise à l'échelle gérée pour le fournisseur de capacité.
4. Associez le fournisseur de capacité à un cluster.
5. Exécutez une tâche ou créez un service avec une stratégie de fournisseur de capacité qui utilise le fournisseur de capacité.

   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.
6. (Facultatif) Définissez une stratégie de fournisseur de capacité par défaut pour le cluster.

Depuis le 27 mai 2022, Amazon ECS ne crée plus de plan de mise AWS Auto Scaling à l'échelle pour les fournisseurs de capacité nouvellement créés. Amazon ECS utilise plutôt la stratégie de dimensionnement de suivi cible jointe au groupe Auto Scaling pour effectuer un dimensionnement dynamique en fonction de vos spécifications de capacité cible. Pour plus d'informations, consultez Fournisseurs de capacité de groupe Amazon EC2 Auto Scaling.

Cette nouvelle version vous permet d'utiliser un groupe Auto Scaling existant avec une stratégie de dimensionnement à utiliser lors de la création d'un nouveau fournisseur de capacité. Nous vous recommandons de ne pas modifier les ressources de stratégie de dimensionnement ou de plan gérées par ECS. Toutefois, lors de la création de nouvelles ressources pour les fournisseurs de capacités, si vous disposez d'outils personnalisés qui ont apporté des modifications au plan de AWS Auto Scaling dimensionnement, effectuez l'une des opérations suivantes :

- (Recommandé) Mettez à jour votre fournisseur de capacité pour modifier les paramètres de mise à l'échelle gérée Amazon ECS. Pour plus d'informations, consultez UpdateCapacityProvider.
- Mettez à jour la stratégie de mise à l'échelle associée à votre groupe Auto Scaling pour modifier la configuration de suivi cible utilisée. Pour plus d'informations, consultez PutScalingPolicy.

Considérations

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.
- 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 plus d'informations sur l'utilisation et la création de rôles IAM liés aux services Amazon ECS, consultez 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 utiliser la `binpack` stratégie sans que la `spread` stratégie ait un ordre supérieur à la `binpack` 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.

Tenez compte des éléments suivants lorsque vous utilisez la console :

- La fonctionnalité de mise à l'échelle gérée Amazon ECS est activée par défaut. Pour plus d'informations, consultez Comportement de montée en puissance gérée.
- La protection contre la résiliation gérée est désactivée par défaut. Pour plus d'informations, voir la section suivante the section called “Protection contre la résiliation gérée”.
- La protection de la mise à l'échelle horizontale de l'instance Auto Scaling est désactivée par défaut. Pour plus d'informations, consultez Protection contre la mise à l'échelle horizontale d'instance dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.
- Le groupe Auto Scaling utilisé avec votre fournisseur de capacité ne peut pas utiliser de paramètres de pondération d'instance. La pondération d'instance n'est pas prise en charge lorsqu'elle est utilisée avec un fournisseur de capacité Amazon ECS.

Présentation de la mise à l'échelle automatique des clusters

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 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
```

`DesiredCapacity`

Quantité de capacité pour le groupe Auto Scaling.

Amazon ECS réalise le processus d'autoscaling de cluster pour chaque fournisseur de capacité associé à vos clusters. Amazon ECS exécute le processus toutes les minutes.

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 des instances EC2, 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 résilier des instances de conteneur EC2, puis annuler leur enregistrement dans le 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 Protection contre la résiliation gérée. Pour plus d'informations sur la résiliation des instances par les groupes Auto Scaling, veuillez consulter Contrôle des instances Auto Scaling à résilier lors d'une mise à l'échelle horizontale dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Protection contre la résiliation gérée

> **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.

L'autoscaling de cluster Amazon ECS met à l'échelle horizontale le groupe Auto Scaling lorsque la valeur CapacityProviderReservation est inférieure au pourcentage `targetCapacity` que vous avez défini. L'autoscaling de cluster permet de contrôler quelles instances sont résiliées si vous activez la protection contre la résiliation gérée. Lorsque vous utilisez la protection contre la résiliation gérée, Amazon ECS résilie uniquement les instances EC2 qui n'ont pas de tâches Amazon ECS 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.

Lorsque vous utilisez la protection contre la résiliation gérée, Amazon ECS active d'abord l'option de protection contre la mise à l'échelle horizontale d'instance pour les instances EC2 du groupe Auto Scaling. Amazon ECS place ensuite les tâches sur les instances. Lorsque toutes les tâches autres que le démon sont arrêtées sur une instance, Amazon ECS lance le processus de mise à l'échelle horizontale et désactive la protection de la mise à l'échelle horizontale pour l'instance EC2. Le groupe Auto Scaling peut ensuite résilier l'instance.

La protection contre la mise à l'échelle horizontale d'instance Auto Scaling contrôle quelles instances EC2 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 contre la mise à l'échelle horizontale d'instance Auto Scaling, veuillez consulter Utilisation de la protection contre la mise à l'échelle horizontale d'instance dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

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 afin de faciliter le dimensionnement automatique du cluster. Amazon ECS gère la CloudWatch métrique. De cette façon, le groupe Auto Scaling est traité comme un état stable de sorte qu'aucune action de mise à l'échelle ne soit 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 valeur de `targetCapacity` inférieure à 100 % représente la quantité de capacité libre (instances Amazon EC2) devant 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. Pour éviter ce comportement, n'utilisez pas la stratégie `spread` en même temps que la stratégie `binpack`.

Tenez compte des points suivants lorsque vous utilisez la protection de résiliation gérée avec la console :

- Par défaut, la protection gérée contre les résiliations est désactivée pour les nouveaux fournisseurs de capacité que vous créez.
- La console n'active pas la protection de mise à l'échelle des instances du groupe Auto Scaling sélectionné. La protection de la mise à l'échelle horizontale de l'instance Auto Scaling est désactivée par défaut. 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 Protection contre la mise à l'échelle horizontale d'instance dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.
- 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.

Comportement de montée en puissance gérée

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. Le comportement de montée en puissance est décrit plus en détail ci-dessous.

1. Amazon ECS sélectionne un fournisseur de capacité pour chaque tâche en suivant la stratégie du fournisseur de capacité définie par le service, par la tâche autonome ou 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.

2. Regroupez toutes les tâches de provisionnement pour ce fournisseur de capacité de façon à ce chaque groupe ait exactement les mêmes besoins en ressources.
3. 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 vCPU, mémoire, interfaces réseau Elastic (ENI), ports et GPU. 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, veuillez consulter Caractérisation de votre application dans le Guide des bonnes pratiques 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 `PENDING`.
   >
   > Pour éviter que les tâches ne restent à l'état `PENDING`, 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.

4. 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 des besoins en vCPU, mémoire, interfaces réseau Elastic (ENI), ports et GPU des tâches. Elle prend également en compte la disponibilité des ressources des instances Amazon EC2. 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`.
5. Amazon ECS publie la `CapacityProviderReservation` métrique CloudWatch par rapport à `minimumScalingStepSize` si l'une des conditions suivantes est le cas. Soit le nombre maximal

d'instances calculé est inférieur à la taille minimale de l'étape de mise à l'échelle. Soit la plus faible des valeurs entre la `maximumScalingStepSize` et le nombre maximal d'instances calculé.

6. 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.

   Vous pouvez définir la `targetCapacity` lorsque vous créez le groupe Auto Scaling ou que vous modifiez la valeur après la création du groupe. La valeur par défaut est 100 %.

7. Le groupe Auto Scaling lance des instances EC2 supplémentaires. Pour éviter le surprovisionnement, Auto Scaling s'assure que la capacité des instances EC2 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 :

1. 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.
2. 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 plus d’informations, consultez Comportement de montée en puissance gérée.
3. 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.

4. 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'instance EC2 puisse lancer et démarrer l'agent Amazon ECS (afin d'éviter le surprovisionnement).
5. 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.
6. 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 .
7. 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.
8. Si un service Amazon ECS ou l'API `run-task` lance une tâche et que les instances de conteneur du fournisseur de capacité ne disposent pas de ressources suffisantes pour démarrer la tâche, Amazon ECS limite le nombre de tâches avec cet état pour chaque cluster et empêche toute tâche de dépasser cette limite. Pour plus d'informations, consultez 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. 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 la résiliation gérée par le fournisseur de capacité, le groupe Auto Scaling sélectionne les instances EC2 à l'aide de la politique de résiliation de groupe Auto Scaling et résilie les instances jusqu'à ce que le nombre d'instances EC2 atteigne la `DesiredCapacity`. 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 résilier les instances EC2. L'enregistrement des instances de conteneur est ensuite annulé du cluster.

Considérations relatives aux mises à l'échelle horizontale

Tenez compte des points suivants pour le processus de mise à l'échelle horizontale :

- Lorsqu'aucune tâche autre que le démon n'est en cours d'exécution, les instances de conteneur Amazon ECS sont considérées comme disponibles pour une 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.

Mise à jour de la façon dont Amazon ECS crée des ressources de mise à l'échelle automatique d'un cluster

Depuis le 27 mai 2022, Amazon ECS a modifié sa façon de gérer les ressources qui facilitent le dimensionnement automatique des clusters. Pour simplifier l'expérience, Amazon ECS n'a plus besoin de plan de AWS Auto Scaling dimensionnement lorsque vous activez le dimensionnement géré pour un fournisseur de capacité de groupe Auto Scaling.

> **Important**
>
> Cette modification n'a aucun impact fonctionnel sur les flux de travail d'autoscaling de votre cluster, ni sur les prix ou les performances.

Fournisseurs de capacités créés avant le 27 mai 2022

Les fournisseurs de capacités, créés avant le 27 mai 2022 et qui utilisent les plans de mise à l'échelle AWS Auto Scaling , continueront à fonctionner comme auparavant.

Examinez les considérations suivantes :

- Nous vous déconseillons de mettre à jour ou de supprimer le plan de `ECS-managed` AWS Auto Scaling dimensionnement ou les ressources de politique de dimensionnement associés à vos fournisseurs de capacité.
- Vous pouvez accéder à la ressource de plan de mise à l'échelle pour les clusters sur la console Auto Scaling, ainsi que par les `describe-clusters` en pièces jointes. Pour plus d'informations, consultez la documentation de l'API DescribeClusters.
- Vous ne pouvez pas ajouter de politiques de dimensionnement au groupe Auto Scaling qui agit en tant que fournisseur de capacité du cluster.
- Le nombre de plans Auto Scaling pour chaque compte est limité. Pour plus d'informations, consultez Quotas pour vos plans de mise à l'échelle dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Fournisseurs de capacités créés le 27 mai 2022 ou après

Depuis le 27 mai 2022, Amazon ECS ne crée plus de plan de mise AWS Auto Scaling à l'échelle pour les fournisseurs de capacité nouvellement créés. Amazon ECS utilise plutôt la stratégie de

dimensionnement de suivi cible jointe au groupe Auto Scaling pour effectuer un dimensionnement dynamique en fonction de vos spécifications de capacité cible. Pour plus d'informations, consultez Fournisseurs de capacité de groupe Amazon EC2 Auto Scaling.

Cette nouvelle version vous permet d'utiliser un groupe Auto Scaling existant avec une stratégie de dimensionnement à utiliser lors de la création d'un nouveau fournisseur de capacité. Nous vous recommandons de ne pas modifier les ressources de stratégie de dimensionnement ou de plan gérées par ECS. Toutefois, lors de la création de nouvelles ressources de fournisseur de capacités, si vous disposez d'outils personnalisés qui ont apporté des modifications au plan de mise à l'échelle AWS Auto Scaling , effectuez l'une des actions suivantes :

- (Recommandé) Mettez à jour votre fournisseur de capacité pour modifier les paramètres de mise à l'échelle gérée Amazon ECS. Pour plus d'informations, consultez UpdateCapacityProvider.
- Mettez à jour la stratégie de mise à l'échelle associée à votre groupe Auto Scaling pour modifier la configuration de suivi cible utilisée. Pour plus d'informations, consultez PutScalingPolicy.

Activation de l'autoscaling de cluster

Vous pouvez utiliser le AWS CLI pour activer le dimensionnement automatique du cluster.

Avant de commencer, créez un groupe Auto Scaling et un fournisseur de capacité. Pour plus d'informations, consultez the section called “Fournisseurs de capacité de groupe Amazon EC2 Auto Scaling”.

Associez le fournisseur de capacité au cluster.

Effectuez les étapes suivantes pour associer le fournisseur de capacité au 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 la section `put-cluster-capacity-providers` dans la référence des commandes 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 lancement EC2, incluez le nom du groupe Auto Scaling dans la demande. Pour plus d'informations, consultez la section `put-cluster-capacity-providers` dans la référence des commandes 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 la section `describe-clusters` dans la référence des commandes AWS CLI .

```
aws ecs describe-clusters \
  --cluster ClusterName \
  --include ATTACHMENTS
```

Activez la mise à l'échelle gérée pour le fournisseur de capacité

Effectuez les étapes suivantes pour activer la mise à l'échelle gérée pour le fournisseur de capacité.

- 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 la section `update-capacity-provider` dans la référence des commandes AWS CLI .

```
aws ecs update-capacity-provider \
  --capacity-providers CapacityProviderName \
  --auto-scaling-group-provider managedScaling=ENABLED
```

Désactivation du cluster Auto Scaling

Vous pouvez utiliser le AWS CLI pour désactiver le dimensionnement automatique du cluster.

Pour désactiver la mise à l'échelle automatique du cluster pour un cluster, vous pouvez dissocier le fournisseur de capacité avec la mise à l'échelle gérée activée du cluster ou en mettant à jour le fournisseur de capacité pour désactiver la mise à l'échelle gérée.

Dissociez le fournisseur de capacité du cluster.

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 la section `put-cluster-capacity-providers` dans la référence des commandes 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 la section `put-cluster-capacity-providers` dans la référence des commandes 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 la section `update-capacity-provider` dans la référence des commandes AWS CLI .

```
aws ecs update-capacity-provider \
  --capacity-providers CapacityProviderName \
  --auto-scaling-group-provider managedScaling=DISABLED
```

Gestion du cluster

Vous trouverez ci-après des concepts généraux sur les clusters Amazon ECS.

- Amazon ECS crée un cluster par défaut. Vous pouvez créer d'autres clusters dans un compte afin de séparer vos ressources.
- Les clusters sont Région AWS spécifiques.
- Les états possibles dans lesquels un cluster peut se trouver sont les 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. Cependant, cela est susceptible de changer à terme. Il est donc important de ne pas compter sur la persistance des clusters `INACTIVE`.
- Un cluster peut contenir une combinaison de tâches hébergées sur AWS Fargate des instances Amazon EC2 ou des instances externes. Les tâches peuvent être exécutées sur l'infrastructure Fargate ou EC2 en tant que type de lancement ou stratégie de fournisseur de capacité. Si vous

utilisez EC2 comme type de lancement, ECS ne suit pas et ne met pas la capacité des groupes Amazon EC2 Auto Scaling à l'échelle. 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. Toutefois, lorsque vous spécifiez une stratégie de fournisseur de capacité, celle-ci peut contenir l'un ou l'autre de ces types, mais pas les deux. Pour plus d'informations, consultez Fournisseurs de capacité Amazon ECS.
- Pour les tâches qui utilisent le type de lancement EC2 ou les fournisseurs de capacité de groupe Auto Scaling, les clusters peuvent contenir plusieurs types d'instances de conteneur différents. Toutefois, chaque instance de conteneur peut uniquement être enregistrée dans un cluster à la fois.
- Des stratégies IAM personnalisées peuvent être créées pour autoriser ou restreindre l'accès des utilisateurs à des clusters spécifiques. Pour plus d'informations, consultez la section Exemples de cluster dans Exemples de politiques basées sur l'identité pour Amazon Elastic Container Service.
- Les clusters comportant des tâches Fargate peuvent être mis à l'échelle à l'aide de Service Auto Scaling. Pour plus d'informations, consultez 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 plus d'informations, consultez Service Connect*.
- Si vous utilisez des instances EC2, la capacité du cluster peut être localisée dans l'une des ressources AWS suivantes :

  Pour plus d'informations sur l'utilisation de ces ressources avec Amazon ECS, veuillez consulter Applications Amazon ECS dans des sous-réseaux partagés, des zones Locales et des zones de longueur d'onde.

  - Zones de disponibilité
  - Local Zones
  - Zones Wavelength
  - AWS Outposts

Création d'un cluster pour le type de lancement Fargate et External à l'aide de la console

Vous pouvez créer un cluster Amazon ECS à l'aide de la console Amazon ECS. Avant de commencer, veillez achever les étapes de Configurer l'utilisation d'Amazon ECS et affectez l'autorisation IAM appropriée. Pour plus d'informations, consultez the section called "Exemples de cluster". 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.

- Crée un espace de noms par défaut portant le même nom que le cluster. AWS Cloud Map 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 plus d'informations, consultez Interconnexion des services.

Vous pouvez modifier les options suivantes :

- Modifiez l'espace de noms par défaut associé 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. L'espace de noms par défaut porte le même nom que le cluster. Pour plus d'informations, consultez Interconnexion des services.
- Configuration du cluster pour les instances externes
- 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 plus d'informations, consultez the section called "Surveillez les conteneurs Amazon ECS à l'aide de Container Insights".
- Ajoutez des balises pour vous aider à identifier vos clusters.

Pour créer un nouveau cluster (console Amazon ECS)

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 des instances externes (Amazon ECS Anywhere) pour votre capacité. Développez Infrastructure, décochez AWS Fargate (sans serveur), puis sélectionnez Instances externes utilisant ECS Anywhere.
7. (Facultatif) Pour activer Container Insights, développez Monitoring (Surveillance), puis activez Use Container Insights (Utiliser Container Insights).
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

Si vous utilisez des instances externes, vous devez enregistrer les instances auprès du cluster. Pour plus d’informations, consultez Enregistrement d'une instance externe dans un cluster.

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
- Exécuter une application en tant que tâche Amazon ECS
- Création d'un service à l'aide de la console

Création d'un cluster pour le type de lancement Amazon EC2 à l'aide de la console

Vous pouvez créer un cluster Amazon ECS à l'aide de la console. Avant de commencer, veillez achever les étapes de Configurer l'utilisation d'Amazon ECS et affectez l'autorisation IAM appropriée. Pour plus d'informations, consultez the section called "Exemples de cluster". 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 des instances Amazon EC2 lors de la création du cluster ou enregistrer des instances supplémentaires avec le cluster après sa création.

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.
- Modifiez l'espace de noms par défaut associé 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. L'espace de noms par défaut porte le même nom que le cluster. Pour plus d'informations, consultez Interconnexion des services.
- 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 plus d'informations, consultez the section called "Surveillez les conteneurs Amazon ECS à l'aide de Container Insights".

- Ajoutez des balises pour vous aider à identifier vos clusters.

Options de groupe Auto Scaling

Lorsque vous utilisez des instances Amazon EC2, 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 empêche la résiliation des instances Amazon EC2 d'un groupe Auto Scaling contenant des tâches lors d'une action de mise à l'échelle horizontale. 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 de plus amples informations sur comment créer des clés SSH, veuillez consulter Paires de clés Amazon EC2 et instances Linux dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.
- 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 pile AWS CloudFormation . 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. Cette option ajoute automatiquement le bloc d'adresses CIDR IPv4 0.0.0.0/0 et le bloc d'adresses CIDR IPv6 ::/0.

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 plus d’informations, consultez the section called “Exemples de cluster”.

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.

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 instances Amazon EC2 à votre cluster, développez Infrastructure, désélectionnez AWS Fargate (sans serveur), puis sélectionnez Instances Amazon EC2. 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, voir Stratégies d'allocation pour les instances Spot dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.
      - Pour Operating system/Architecture (Système d'exploitation/architecture), choisissez l'AMI optimisée pour Amazon ECS pour les instances de groupe Auto Scaling.
      - Pour EC2 instance type (Type d'instance EC2), 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.

- Pour le rôle d'instance EC2, choisissez un rôle d'instance de conteneur existant ou créez-en un nouveau.

  Pour plus d'informations, consultez 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 instances Amazon EC2, 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 conteneur Amazon EC2, dans Attribuer automatiquement l'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) Pour activer Container Insights, développez Monitoring (Surveillance), puis activez Use Container Insights (Utiliser 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) 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.
11. 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
- Exécuter une application en tant que tâche Amazon ECS
- Création d'un service à l'aide de la console

Mise à jour d'un cluster à l'aide de la console

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.

Les fournisseurs de capacité constituent une alternative aux types de lancement. Pour plus d'informations, consultez Concepts des fournisseurs de capacité.

- 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 plus d'informations, consultez the section called "Surveillez les conteneurs Amazon ECS à l'aide de Container Insights".
- Ajoutez des étiquettes pour vous aider à identifier vos clusters.

Pour mettre à jour le cluster (console Amazon ECS)

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur la page Cluster : ***nom***, 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 avec la console

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 plus d'informations, consultez the section called “Suppression d'un service”.
- Arrêtez toutes les tâches en cours d'exécution. Pour plus d'informations, consultez the section called “Arrêter une tâche Amazon ECS”.
- Annulez l'enregistrement de toutes les instances de conteneur enregistrées dans le cluster. Pour plus d'informations, consultez 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 .

Pour supprimer un cluster (console Amazon ECS)

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 la zone de confirmation, saisissez delete ***cluster name*** (supprimer le nom du cluster).

Création d'un fournisseur de capacité pour un cluster à l'aide de la console

Une fois la création du cluster terminée, vous pouvez créer un nouveau fournisseur de capacité (groupe Auto Scaling) pour le type de lancement EC2.

Avant de créer le fournisseur de capacité, vous devez créer un groupe Auto Scaling. Pour plus d'informations, voir Groupes Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Pour créer un fournisseur de capacité pour le cluster (console Amazon ECS)

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur la page Cluster : ***nom***, choisissez Infrastructure, puis choisissez Create (Créer).
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 résiliation de l'instance EC2 exécutant des tâches Amazon ECS, sélectionnez Turn on scaling protection (Activer la protection du 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.

Mise à jour d'un fournisseur de capacité pour un cluster à l'aide de la console

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur la page Cluster : ***nom***, choisissez Infrastructure, puis choisissez Update (Mettre à jour).
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 empêcher la résiliation des instances EC2 exécutant des tâches Amazon ECS, sélectionnez Activer la protection du 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.

Suppression d'un fournisseur de capacité pour un cluster à l'aide de la console

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 de groupe Auto Scaling passe à

l'état `INACTIVE`. 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 l'option Forcenew Deployment pour vous assurer que toutes les tâches utilisant la capacité d'instance Amazon EC2 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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur la page Cluster : ***nom***, choisissez Infrastructure, le groupe Auto Scaling, puis choisissez Delete (Supprimer).
5. Dans la zone de confirmation, saisissez delete ***Auto Scaling group name*** (supprimer le nom du groupe Auto Scaling).
6. Sélectionnez Delete (Supprimer).

Création de capacité

Une instance de conteneur Amazon ECS est une instance Amazon EC2 qui exécute l'agent de conteneur Amazon ECS et qui a été enregistrée dans un cluster Amazon ECS. Lorsque vous exécutez des tâches avec Amazon ECS à l'aide du type de lancement EC2, External ou d'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.

Amazon ECS fournit les AMI optimisées pour Amazon ECS qui sont préconfigurées avec les exigences et les recommandations pour exécuter vos charges de travail de conteneur. Nous vous recommandons d'utiliser l'AMI Amazon Linux 2023 optimisée pour Amazon ECS pour vos instances Amazon EC2, sauf si votre application nécessite des instances Amazon EC2 basées sur un GPU, un système d'exploitation spécifique ou une version de Docker qui n'est pas encore disponible dans

cette AMI. Pour plus d'informations sur les instances Amazon Linux 2 et Amazon Linux 2023, veuillez consulter Comparaison entre Amazon Linux 2 et Amazon Linux 2023 dans le Guide de l'utilisateur Amazon Linux 2023 (langue française non garantie).

Bien que vous puissiez créer votre propre AMI d'instance Amazon EC2 répondant aux spécifications de base nécessaires pour exécuter vos charges de travail conteneurisées sur Amazon ECS, les AMI optimisées pour Amazon ECS sont préconfigurées et testées sur Amazon ECS par des ingénieurs. AWS C'est le moyen le plus simple de démarrer et de faire fonctionner AWS rapidement vos conteneurs.

Vous pouvez exécuter Linux ou Windows sur vos instances de conteneur. Vous pouvez choisir le type d'instance en fonction des exigences de votre application, telles que l'architecture du processeur, le débit réseau et l'architecture. 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. Vous pouvez lancer plusieurs tâches en même temps avec des instances de plus grande taille. Vous pouvez effectuer une mise à l'échelle plus fine pour réduire les coûts avec des instances plus petites. Il n'est pas nécessaire de choisir un seul type d'instance Amazon EC2 adapté à toutes les applications de votre cluster. Vous pouvez créer plusieurs groupes Auto Scaling avec différents types d'instances adaptés à toutes les applications. Pour plus d'informations, veuillez consulter Instances Amazon EC2.

Pour déterminer les types d'instances que vous pouvez utiliser, commencez par éliminer les types d'instances ou les 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. Cependant, vous devez également tenir compte d'autres exigences. Par exemple, considérez l'architecture du processeur, le débit du réseau et déterminez si le stockage d'instance est une exigence. Examinez ensuite la quantité de processeur et de mémoire fournie par chaque type d'instance. 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.

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 monter en puissance de manière plus précise afin de réduire les coûts. Il n'est pas nécessaire de choisir un seul type d'instance Amazon EC2 adapté à toutes les applications de votre cluster. Au lieu de cela, vous pouvez créer plusieurs groupes Auto Scaling. Chaque groupe peut avoir un type d'instance différent. Vous pouvez ensuite créer un fournisseur de capacité Amazon EC2 pour chacun de ces groupes. Enfin, dans la stratégie de fournisseur de capacité de votre service

et de votre tâche, vous pouvez sélectionner le fournisseur de capacité qui répond le mieux à ses besoins. Pour plus d'informations, veuillez consulter Instances Amazon EC2.

Suivez les instructions suivantes pour mettre en réseau vos instances :

- Nous vous recommandons de lancer vos instances de conteneur à l'intérieur d'un VPC, car Amazon VPC permet de mieux contrôler votre réseau et offre des capacités de configuration plus approfondies. Pour plus d'informations, consultez Amazon EC2 et Amazon Virtual Private Cloud dans le Amazon EC2 Guide de l'utilisateur pour les instances Linux.
- Si les conteneurs associés à vos tâches nécessitent une connectivité externe, vous pouvez mapper leurs ports réseau à des ports sur l'instance de conteneur Amazon ECS hôte afin qu'ils soient accessibles depuis Internet. Votre groupe de sécurité d'instance de conteneur doit autoriser l'accès entrant sur les ports que vous souhaitez exposer. Pour plus d'informations, veuillez consulter Créer un groupe de sécurité dans le Guide de mise en route Amazon VPC.
- Les instances de conteneur ont besoin de communiquer avec le point de terminaison de service Amazon ECS service. 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, veuillez consulter 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 Configuration du proxy HTTP pour les instances de conteneur Linux dans ce guide. Pour plus d’informations, consultez the section called “Créer un Virtual Private Cloud”.

Spot instances

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 Spot d'Amazon EC2. Dans ce 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 résiliées 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'instances Amazon EC2 : lorsque vous utilisez des stratégies 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 Groupes Auto Scaling avec types d’instance et options d’achat multiples dans le Guide de l’utilisateur Amazon EC2 Auto Scaling.
- Utilisez la stratégie d'allocation Spot 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 optimisée en termes de 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.

Drainage des instances Linux Spot

Amazon EC2 résilie, arrête ou met en veille prolongée votre instance Spot lorsque le prix Spot dépasse le prix maximum de votre demande ou lorsque la capacité n'est plus disponible. Amazon EC2 fournit un avis d'interruption de deux minutes pour les actions de résiliation et d'arrêt des instances Spot. 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, ECS reçoit l'avis d'interruption d'instance Spot et bascule l'instance à l'état `DRAINING`.

> **Important**
>
> Amazon ECS ne reçoit pas d'avis de la part d'Amazon EC2 lorsque les instances sont supprimées par le rééquilibrage de la capacité Auto Scaling. Pour plus d'informations, consultez Rééquilibrage de la capacité Amazon EC2 Auto Scaling.

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 et doit être activé manuellement. Pour activer le drainage d'une instance Spot pour une nouvelle instance de conteneur, lors du lancement de l'instance de conteneur, ajoutez le script suivant dans le champ Données utilisateur, en le *`MyCluster`*remplaçant 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 plus d’informations, consultez 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
     ```

   - Pour l'AMI Amazon Linux optimisée pour Amazon ECS :

     ```
     sudo stop ecs && sudo start 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 plus d'informations, consultez the section called “Introspection des conteneurs”.

   ```
   curl http://localhost:51678/v1/metadata
   ```

Drainage des instances Windows Spot

Amazon EC2 résilie, arrête ou met en veille prolongée votre instance Spot lorsque le prix Spot dépasse le prix maximum de votre demande ou lorsque la capacité n'est plus disponible. Amazon EC2 communique un avis d'interruption d'instance Spot, qui donne à l'instance un avertissement deux minutes avant qu'elle soit interrompue. 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 devez définir le paramètre `ECS_ENABLE_SPOT_INSTANCE_DRAINING` avant de démarrer l'agent de conteneur. Utilisez les commandes suivantes pour activer manuellement le drainage des instances Spot. 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 plus d'informations, consultez the section called “Lancement d'une instance de conteneur”.

Instances Linux

Vous pouvez utiliser des instances EC2 avec les systèmes d'exploitation Linux suivants pour exécuter vos applications :

- Amazon Linux : il s'agit d'un système d'exploitation polyvalent.
- Bottlerocket : Bottlerocket est un système d'exploitation open source basé sur Linux 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 inclut uniquement le nombre minimum de packages requis pour faire fonctionner les conteneurs. Cela améliore l'utilisation des ressources, réduit la surface d'attaque de sécurité et contribue à réduire les frais de gestion. Pour plus d'informations sur les fonctionnalités de sécurité et les instructions, consultez les fonctionnalités de sécurité et les directives de sécurité sur le GitHub site Web.

Amazon ECS fournit les AMI optimisées pour Amazon ECS qui sont préconfigurées avec les exigences et les recommandations pour exécuter vos charges de travail de conteneur. Nous vous recommandons d'utiliser l'AMI Amazon Linux 2023 optimisée pour Amazon ECS pour vos instances Amazon EC2, sauf si votre application nécessite des instances Amazon EC2 basées sur un GPU, un système d'exploitation spécifique ou une version de Docker qui n'est pas encore disponible dans cette AMI. Pour plus d'informations sur les instances Amazon Linux 2 et Amazon Linux 2023, veuillez

consulter Comparaison entre Amazon Linux 2 et Amazon Linux 2023 dans le Guide de l'utilisateur Amazon Linux 2023 (langue française non garantie).

Bien que vous puissiez créer votre propre AMI d'instance Amazon EC2 répondant aux spécifications de base nécessaires pour exécuter vos charges de travail conteneurisées sur Amazon ECS, les AMI optimisées pour Amazon ECS sont préconfigurées et testées sur Amazon ECS par des ingénieurs. AWS C'est le moyen le plus simple de démarrer et de faire fonctionner AWS rapidement vos conteneurs.

Une spécification d'une instance de conteneur Amazon ECS comprend les trois composants suivants :

Obligatoire

- Une distribution Linux exécutant au moins la version 3.10 du noyau Linux.
- L'agent de conteneur Amazon ECS (de préférence la version la plus récente). Pour plus d'informations, consultez Mise à jour de l'agent de conteneur Amazon ECS.
- Un démon Docker exécutant au moins la version 1.9.0 et toutes les spécifications relatives à l'exécution de Docker. Pour plus d'informations, consultez la section Check runtime dependencies dans la documentation Docker.

> **Note**
>
> Pour une expérience optimale, nous vous recommandons d'utiliser la version de Docker fournie et testée avec la version de l'agent de conteneur Amazon ECS que vous utilisez.

Recommandée

- Un processus d'initialisation et un processus de surveillance qui permet d'exécuter et de surveiller l'agent de conteneur Amazon ECS. Les AMI optimisées pour Amazon ECS utilisent le RPM `ecs-init` pour gérer l'agent. Pour plus d'informations, consultez le `ecs-init`projet sur GitHub.

Rubriques

- AMI optimisée pour Amazon ECS
- AMI Bottlerocket optimisées pour Amazon ECS
- Installation de l'agent de conteneur Amazon ECS

AMI optimisée pour Amazon ECS

Amazon ECS fournit les AMI optimisées pour Amazon ECS qui sont préconfigurées avec les exigences et les recommandations pour exécuter vos charges de travail de conteneur. Nous vous recommandons d'utiliser l'AMI Amazon Linux 2023 optimisée pour Amazon ECS pour vos instances Amazon EC2, sauf si votre application nécessite des instances Amazon EC2 basées sur un GPU, un système d'exploitation spécifique ou une version de Docker qui n'est pas encore disponible dans cette AMI. Pour plus d'informations sur les instances Amazon Linux 2 et Amazon Linux 2023, veuillez consulter Comparaison entre Amazon Linux 2 et Amazon Linux 2023 dans le Guide de l'utilisateur Amazon Linux 2023 (langue française non garantie). 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.

Les variantes Linux de l'AMI optimisée pour Amazon ECS utilisent l'AMI Amazon Linux 2 comme base. 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.

Nous vous recommandons d'utiliser une AMI avec le noyau Linux 5.10 car le noyau Linux 4.14 est arrivé end-of-life le 10 janvier. 2024.

Les variantes suivantes de l'AMI optimisée pour Amazon ECS sont disponibles pour vos instances Amazon EC2.

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. Recommandée dans la plupart des cas pour le lancement de vos instances Amazon EC2 pour vos charges de travail	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

Système d'exploitation	AMI	Description	Configuration du stockage
		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).	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.
Amazon Linux 2023 (arm64)	AMI Amazon Linux 2023 (arm64) optimisée pour Amazon ECS	Basée sur Amazon Linux 2023, cette AMI est recommandée pour le lancement de vos instances Amazon EC2 à technologie de processeurs Graviton/Graviton 2 AWS basés sur Arm pour vos charges de travail Amazon ECS. Pour plus d'informations, consultez Instances polyvalentes dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux. L'AMI Amazon Linux 2023 (arm64) optimisée pour Amazon ECS n'est	

Système d’exploitation	AMI	Description	Configuration du stockage
		pas fournie avec le préinstallé. AWS CLI	
Amazon Linux 2023 (Neuron)	AMI Amazon Linux 2023 (Neuron) optimisée pour Amazon ECS	Basée sur Amazon Linux 2023, cette AMI est recommandée pour le lancement de vos instances Amazon EC2 Inf1. Il est préconfiguré avec les pilotes AWS Inferentia et le moteur d'exécution AWS Neuron pour Docker, ce qui facilite l'exécution des charges de travail d'inférence liées au machine learning sur Amazon ECS. Pour plus d’informations, consultez Utilisation de AWS Neuron sur Amazon Linux 2 sur Amazon ECS. L'AMI Amazon Linux 2023 (Neuron) optimisée pour Amazon ECS n'est pas fournie avec le AWS CLI préinstallé.	

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 à utiliser lorsque vous lancez vos instances Amazon EC2 et que 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 Linux 2 optimisées pour Amazon ECS (AMI Amazon Linux 2 optimisée pour Amazon ECS, AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS et AMI optimisée pour GPU Amazon ECS) sont livrées avec un seul volume racine 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 2 optimisée pour Amazon ECS
	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.	
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 instances Amazon EC2, qui sont alimentées par des processeurs AWS Graviton/Graviton	

Système d'exploitation	AMI	Description	Configuration du stockage
		2 basés sur 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 Instances polyvalentes dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux. L'AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS n'est pas fournie avec le préinstallé. AWS CLI	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
	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 instances Amazon EC2, qui sont alimentées par des processeurs AWS Graviton/Graviton 2 basés sur 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	

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 basées sur le GPU Amazon EC2 avec le noyau Linux 5.10 pour vos charges de travail Amazon ECS. Elle est livrée préconfigurée avec les pilotes du noyau NVIDIA et une exécution de processeur graphique Docker qui font s'exécuter les applications qui tirent parti des GPU sur Amazon ECS. Pour plus d’informations, consultez Utilisation des GPU sur Amazon ECS.	

Système d’exploitation	AMI	Description	Configuration du stockage
	AMI optimisée pour GPU Amazon ECS	Basée sur Amazon Linux 2, il est recommandé d'utiliser cette AMI lors du lancement de vos instances basées sur le GPU Amazon EC2 avec le noyau Linux 4.14 pour vos charges de travail Amazon ECS. Elle est livrée préconfigurée avec les pilotes du noyau NVIDIA et une exécution de processeur graphique Docker qui font s'exécuter les applications qui tirent parti des GPU sur Amazon ECS. Pour plus d’informations, consultez Utilisation des GPU sur Amazon ECS.	

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 Inf1, Trn1 ou Inf2 d'Amazon EC2. 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 plus d’informations, consultez Utilisation de AWS Neuron sur Amazon Linux 2 sur Amazon ECS. L'AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS n'est pas fournie avec le AWS CLI préinstallé.	

Système d’exploitation	AMI	Description	Configuration du stockage
	AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS	Basée sur Amazon Linux 2, cette AMI est destinée aux instances Inf1, Trn1 ou Inf2 d'Amazon EC2. 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 plus d’informations, consultez Utilisation de AWS Neuron sur Amazon Linux 2 sur Amazon ECS. L'AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS n'est pas fournie avec le AWS CLI préinstallé.	

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 source Amazon Linux 2 ou de l'AMI Amazon Linux 2023 pour chaque variante en interrogeant l'API de Systems Manager Parameter Store. Pour plus d'informations, consultez Extraction des métadonnées d'AMI optimisée 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 les anciens emballages, consultez le journal des modifications d'Amazon ecs-init sur GitHub

Extraction des métadonnées d'AMI optimisée 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.

L'ID d'AMI, le nom d'image, le système d'exploitation, la version de l'agent de conteneur, le nom de l'image source et la version d'exécution des AMI optimisées pour Amazon ECS peuvent être extraits 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 GetParametersetGetParametersByPath.

> **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.

AMI optimisées pour Amazon ECS Linux

- 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/inf/<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 extrait l'ID d'image de la dernière AMI Amazon Linux 2 optimisée pour Amazon ECS stable 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.

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 .

AMI optimisées pour Amazon ECS Linux

- Pour les AMI Amazon Linux 2023 optimisées pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended --region us-east-1
```

- Pour les AMI Amazon Linux 2023 (arm64) optimisées pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
arm64/recommended --region us-east-1
```

- Pour les AMI Amazon Linux 2 noyau 5.10 optimisées pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/recommended --region us-east-1
```

- Pour les AMI Amazon Linux 2 optimisées pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
recommended --region us-east-1
```

- Pour les AMI Amazon Linux 2 noyau 5.10 (arm64) optimisées pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/arm64/recommended --region us-east-1
```

- Pour les AMI Amazon Linux 2 (arm64) optimisées pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/arm64/
recommended --region us-east-1
```

- Pour les AMI du noyau 5.10 optimisées pour le GPU Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/gpu/recommended --region us-east-1
```

- Pour les AMI optimisées pour le GPU Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

- Pour les AMI du noyau Amazon Linux 2 (Neuron) 5.10 optimisées 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 les AMI Amazon Linux 2 (Neuron) optimisées 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/recommended/image_id
```

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.

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 constituent la source de vérité pour les builds d'AMI optimisés pour Amazon ECS. Vous pouvez donc suivre le GitHub référentiel pour surveiller les modifications apportées à nos AMI. 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
```

AMI Bottlerocket optimisées pour Amazon ECS

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 5.1.10 :

- `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.

- Bottlerocket prend en charge les instances Amazon EC2 avec processeurs `x86_64` et `arm64`. Il n'est pas recommandé d'utiliser une AMI Bottlerocket avec les instances Amazon EC2 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'agent AWS Systems Manager que vous pouvez utiliser pour exécuter des commandes ou démarrer des sessions shell sur les instances Bottlerocket d'Amazon EC2. 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.
- Les AMI Bottlerocket ne prennent pas non plus en charge les services et fonctionnalités suivants :
  - ECS Anywhere
  - Service Connect
  - Amazon EFS en mode chiffré et en mode réseau `awsvpc`
  - Accélérateur Elastic Inference

Récupération d'une AMI Bottlerocket optimisée pour Amazon ECS

Vous pouvez récupérer l'identifiant Amazon Machine Image (AMI) pour les AMI optimisées pour Amazon ECS en interrogeant l'API AWS Systems Manager Parameter Store. Grâce à ce paramètre, vous n'avez pas besoin de rechercher manuellement les ID d'AMI optimisées pour Amazon ECS. 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.

Récupération de la variante d'AMI Bottlerocket **`aws-ecs-2`**

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#
```

Récupération de la variante d'AMI Bottlerocket **aws-ecs-2-nvidia**

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#
```

Récupération de la variante d'AMI Bottlerocket **`aws-ecs-1`**

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
```

Récupération de la variante d'AMI Bottlerocket **`aws-ecs-1-nvidia`**

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 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 Bottlerocket et Amazon ECS GitHub sur le AWS site du blog.

Pour plus d'informations sur le lancement d'une instance Bottlerocket, voir Lancement d'une Bottlerocket instance

Lancement d'une Bottlerocket instance

Vous pouvez lancer une Bottlerocket instance afin d'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 d'une AMI Bottlerocket optimisée pour Amazon ECS” 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 :
   - Remplacez le *`sous-réseau`* par l'ID du sous-réseau privé ou public dans lequel votre instance sera lancée.
   - Remplacez *`bottlerocket_ami`* par l'ID d'AMI de l'étape précédente.
   - Remplacez *`t3.large`* par le type d'instance que vous souhaitez utiliser.
   - Remplacez *`region`* par le code de la 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.
   - Remplacez *`region`* par le code de votre 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.

Installation de l'agent de conteneur Amazon ECS

Si votre instance de conteneur n'a pas été lancée à l'aide d'une AMI optimisée pour Amazon ECS, vous pouvez installer l'agent de conteneur Amazon ECS manuellement à l'aide de l'une des procédures suivantes. L'agent de conteneur Amazon ECS est inclus dans les AMI optimisées pour Amazon ECS et ne requiert pas d'installation.

- Pour les instances Amazon Linux 2 , vous pouvez installer l'agent à l'aide de la commande `amazon-linux-extras`. Pour plus d'informations, consultez Installation de l'agent du conteneur Amazon ECS sur une instance EC2 Amazon Amazon Linux 2.
- Pour les instances AMI Amazon Linux, vous pouvez installer l'agent à l'aide du référentiel Amazon YUM. Pour plus d'informations, consultez Installation de l'agent de conteneur Amazon ECS sur une instance Amazon Linux EC2.
- Pour les instances autres que Amazon Linux, vous pouvez télécharger l'agent à partir d'un des compartiments S3 régionaux ou à partir d'Amazon Elastic Container Registry Public. Si vous téléchargez à partir d'un des compartiments S3 régionaux, vous pouvez éventuellement vérifier la validité du fichier d'agent de conteneur à l'aide de la signature PGP. Pour plus d'informations, consultez Installation de l'agent du conteneur Amazon ECS sur une instance EC2 autre que Amazon Linux.

> **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 processus `cloud-init` n'est pas considéré comme terminé tant que les données utilisateur Amazon EC2 n'ont pas été complètement exécutées. Par conséquent, le démarrage d'Amazon ECS ou de Docker via les données utilisateur Amazon EC2 peut provoquer un blocage. Pour démarrer l'agent de conteneur à l'aide des données utilisateur Amazon EC2, vous pouvez utiliser `systemctl enable --now --no-block ecs.service`.

Installation de l'agent du conteneur Amazon ECS sur une instance EC2 Amazon Amazon Linux 2

Pour installer l'agent de conteneur Amazon ECS sur une instance EC2 Amazon Linux 2 à l'aide de la commande `amazon-linux-extras`, utilisez les étapes suivantes.

Pour installer l'agent du conteneur Amazon ECS sur une instance EC2 Amazon Amazon Linux 2

1. Lancez une instance Amazon Linux 2 EC2 avec un rôle IAM qui autorise l'accès à Amazon ECS. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.
2. Connectez-vous à votre instance.
3. Désactivez le référentiel supplémentaire `docker` Amazon Linux. Le référentiel supplémentaire `ecs` Amazon Linux est fourni avec sa propre version de Docker. Par conséquent, le `docker` supplémentaire doit être désactivé afin d'éviter les conflits potentiels. Vous êtes ainsi sûr d'utiliser systématiquement la version de Docker prévue pour vous par Amazon ECS pour une utilisation avec une version particulière de l'agent de conteneur.

   ```
   [ec2-user ~]$ sudo amazon-linux-extras disable docker
   ```

4. Installez et activez le référentiel supplémentaire `ecs` Amazon Linux.

   ```
   [ec2-user ~]$ sudo amazon-linux-extras install -y ecs; sudo systemctl enable --now
    ecs
   ```

5. (Facultatif) Vous pouvez vérifier si l'agent est en cours d'exécution et consulter des informations sur votre nouvelle instance de conteneur avec l'API d'introspection d'agent. Pour plus d'informations, consultez the section called “Introspection des conteneurs”.

   ```
   [ec2-user ~]$ curl -s http://localhost:51678/v1/metadata | python -mjson.tool
   ```

   > **Note**
   >
   > Si vous obtenez pas de réponse, assurez-vous que vous avez associé le rôle IAM d'instance de conteneur Amazon ECS lors du lancement de l'instance. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.

Installation de l'agent de conteneur Amazon ECS sur une instance Amazon Linux EC2

Pour installer l'agent de conteneur Amazon ECS sur une instance Amazon Linux EC2 à l'aide du référentiel Amazon YUM, procédez comme suit.

Pour installer l'agent de conteneur Amazon ECS sur une instance Amazon Linux EC2

1. Lancez une instance Amazon Linux EC2 avec un rôle IAM qui permet d'accéder à Amazon ECS. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.
2. Connectez-vous à votre instance.
3. Installez le package `ecs-init`. Pour plus d'informations`ecs-init`, consultez le code source sur GitHub.

   ```
   [ec2-user ~]$ sudo yum install -y ecs-init
   ```

4. Lancez le démon Docker.

   ```
   [ec2-user ~]$ sudo service docker start
   ```

   Sortie :

   ```
   Starting cgconfig service:                                 [  OK  ]
   Starting docker:                                       [  OK  ]
   ```

5. Démarrez la tâche upstart `ecs-init`.

   ```
   [ec2-user ~]$ sudo service ecs start
   ```

   Sortie :

   ```
   ecs start/running, process 2804
   ```

6. (Facultatif) Vous pouvez vérifier si l'agent est en cours d'exécution et consulter des informations sur votre nouvelle instance de conteneur avec l'API d'introspection d'agent. Pour plus d'informations, consultez the section called “Introspection des conteneurs”.

   ```
   [ec2-user ~]$ curl -s http://localhost:51678/v1/metadata | python -mjson.tool
   ```

Installation de l'agent du conteneur Amazon ECS sur une instance EC2 autre que Amazon Linux

Pour installer l'agent de conteneur Amazon ECS sur une instance EC2 autre qu'Amazon Linux, vous pouvez télécharger l'agent à partir d'un des compartiments S3 régionaux et l'installer.

> **Note**
>
> Lorsque vous utilisez une AMI Linux autre qu'Amazon, votre instance Amazon EC2 nécessite une prise en charge de `cgroupfs` pour le pilote `cgroup` pour que l'agent Amazon ECS prenne 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 x86_64 (x86_64)

Région	Nom de la région	Fichiers deb init Amazon ECS	Fichiers rpm init Amazon ECS
		Amazon ECS init arm64 (arm64)	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)
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)

Région	Nom de la région	Fichiers deb init Amazon ECS	Fichiers rpm init Amazon ECS
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)
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)

Région	Nom de la région	Fichiers deb init Amazon ECS	Fichiers rpm init Amazon ECS
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)
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 instance Amazon EC2 autre qu'Amazon Linux AMI

1. Lancez une instance Amazon EC2 avec un rôle IAM qui autorise l'accès à Amazon ECS. Pour plus d’informations, consultez 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 `1.9.0`.

```
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 vos variables d'environnement d'agent dans Amazon S3. Elles pourront être téléchargées dans vos instances de conteneur lors

du lancement à l'aide des données utilisateur Amazon EC2. 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 d'instance de conteneur dans Simple Storage Service (Amazon S3) et Authentification de registre privé pour les tâches.

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 de service de métadonnées de l'instance Amazon EC2 (`http://169.254.169.254`) pour les conteneurs lancés 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 plus d'informations, consultez Rôle IAM de tâche.

Cela permet également à l'agent de conteneur de ne pas devoir disputer les connexions et le trafic réseau sur la passerelle `docker0`.

instances Windows

Une instance de conteneur Amazon ECS est une instance Amazon EC2 qui exécute l'agent de conteneur Amazon ECS et qui a été enregistrée dans un cluster Amazon ECS. Lorsque vous exécutez des tâches avec Amazon ECS, les tâches qui utilisent le type de lancement EC2 ou un fournisseur de capacité de groupe Auto Scaling sont placées sur vos instances de conteneur actives.

> **Note**
>
> Les tâches qui utilisent le type de lancement Fargate sont déployées sur une infrastructure gérée par AWS. Cette rubrique ne s'applique donc pas.

Rubriques

- AMI optimisée pour Amazon ECS
- Création de votre propre AMI Windows optimisée pour Amazon ECS

AMI optimisée pour Amazon ECS

Les AMI optimisées pour Amazon ECS sont préconfigurées avec les composants nécessaires pour exécuter des 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 AMI optimisées pour Amazon ECS 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 plus d’informations, consultez the section called “Extraction des métadonnées d'AMI optimisée pour Amazon ECS”.

Vous pouvez vous abonner aux rubriques Amazon SNS de l'AMI Windows pour être averti lorsqu'une nouvelle AMI est publiée ou qu'une version de l'AMI est marquée comme privée. Pour plus d’informations, consultez Abonnement aux notifications de mise à jour des AMI optimisées pour Amazon ECS.

> **Important**
>
> Toutes les variantes AMI optimisées pour ECS produites après le mois d'août migreront de Docker EE (Mirantis) vers Docker CE (projet Moby).
> Afin de s'assurer que les clients disposent des mises à jour de sécurité les plus récentes par défaut, Amazon ECS gère au moins les trois dernières AMI optimisées pour Windows Amazon ECS. Après avoir publié de nouvelles AMI optimisées pour Windows Amazon ECS, Amazon ECS crée des AMI optimisées pour Windows Amazon ECS qui sont privées et 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 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
- AMI de Windows Server 20H2 Core optimisée pour Amazon ECS

> **Important**
>
> 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 2022, Windows Server 2019 et Windows Server 2016 sont des versions Canal de maintenance à long terme. Windows Server 20H2 est une version Canal semi-annuelle. Pour plus d'informations, consultez les informations sur la version de Windows Server.

Mises en garde du conteneur Windows

Voici quelques éléments que vous devez savoir sur les conteneurs Windows Amazon EC2 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 qui utilisent les types de lancements 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 plus d'informations, consultez Considérations relatives à EC2 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 plus d'informations, consultez Configuration supplémentaire des rôles IAM pour les tâches 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 plus d'informations, consultez Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge.
- 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 instance EC2 exécutée sur la même version en utilisant la contrainte de placement : `memberOf(attribute:ecs.os-family == WINDOWS_SERVER_<OS_Release>_<FULL or CORE>)`. Pour plus d'informations, consultez the section called “Extraction des métadonnées d'AMI optimisée pour Amazon ECS”.

Extraction des métadonnées d'AMI optimisée pour Amazon ECS

L'ID d'AMI, le nom d'image, le système d'exploitation, la version de l'agent de conteneur et la version d'exécution des AMI optimisées pour Amazon ECS peuvent être extraits 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 GetParametersset 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 d'API Systems Manager Parameter Store suivants sont obsolètes et ne doivent pas être utilisés pour récupérer les dernières AMI Windows :
>
> - `/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 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 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
```

Abonnement aux notifications de mise à jour des AMI optimisées pour Amazon ECS

AWS fournit deux ARN de rubrique Amazon SNS pour les notifications relatives aux AMI Windows Server. Une rubrique envoie des notifications de mise à jour lorsque de nouvelles AMI Windows Server sont publiées. L'autre rubrique envoie des notifications lorsque les AMI Windows Server précédemment publiées sont rendues privées. Bien que ces rubriques ne soient pas spécifiques aux AMI Windows optimisées pour Amazon ECS, car ces dernières suivent le même calendrier de publication, vous pouvez utiliser ces notifications pour indiquer le moment où les nouvelles AMI Windows optimisées pour Amazon ECS sont mises à jour. Pour plus d'informations sur l'abonnement aux notifications d'AMI Windows, veuillez consulter Subscribing to Windows AMI notifications dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.

> **Note**
>
> Votre utilisateur, ou le rôle qui s'y rattache, doit disposer des autorisations IAM `sns::subscribe` pour pouvoir s'abonner à une rubrique Amazon SNS.

Versions d'AMI optimisée pour Amazon ECS

Consultez les versions actuelles et précédentes des AMI optimisées pour Amazon ECS ainsi que 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 plus d'informations, consultez the section called "Extraction des métadonnées d'AMI optimisée pour Amazon ECS".

Versions d'AMI optimisées pour Amazon ECS Windows

Les onglets suivants affichent une liste des versions d'AMI optimisées pour Amazon ECS Windows.

> **Note**
>
> Pour de plus amples informations sur le référencement du paramètre Systems Manager Parameter Store dans un modèle AWS CloudFormation , veuillez consulter Utilisation de l'AMI optimisée pour Amazon ECS la plus récente recommandée dans un modèle AWS CloudFormation.

> **Important**
>
> Afin de s'assurer que les clients disposent des mises à jour de sécurité les plus récentes par défaut, Amazon ECS gère au moins les trois dernières AMI optimisées pour Windows Amazon ECS. Après avoir publié de nouvelles AMI optimisées pour Windows Amazon ECS, Amazon ECS crée des AMI optimisées pour Windows Amazon ECS qui sont privées et plus anciennes. S'il existe une AMI privée à laquelle vous devez accéder, soumettez un ticket auprès du support Cloud.

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-2024.02.13	`1.81.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2_English-Full-ECS_Optimized-2024.01.09	`1.79.2`	`20.10.23 (Docker CE)`	Public
Windows_Server-2_English-Full-ECS_Optimized-2023.12.12	`1.79.1`	`20.10.23 (Docker CE)`	Public
Windows_Server-2_English-Full-ECS_Optimized-2023.11.14	`1.79.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2022-English-Full-ECS_Optimized-2023.10.11	`1.77.0`	`20.10.21 (Docker CE)`	Public
Windows_Server-2022-English-Full-ECS_Optimized-2023.09.15	`1.75.3`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2023.08.09	`1.74.1`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2022-English-Full-ECS	`1.73.1`	`20.10.21 (Docker CE)`	Privé

AMI Windows Server 2022 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
_Optimized-2023.07.11			
Windows_Server-2022-English-Full-ECS_Optimized-2023.06.13	1.72.0	20.10.21 (Docker CE)	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2023.05.18	1.71.1	20.10.21 (Docker CE)	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2023.04.18	1.70.2	20.10.21 (Docker CE)	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2023.03.21	1.69.0	20.10.21 (Docker CE)	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2023.02.21	1.68.2	20.10.21 (Docker CE)	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2023.01.11	1.68.0	20.10.21 (Docker CE)	Privé

AMI Windows Server 2022 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2022-English-Full-ECS_Optimized-2022.12.14	1.67.2	20.10.21 (Docker CE)	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2022.11.09	1.65.1	20.10.21 (Docker CE)	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2022.10.12	1.64.0	20.10.17 (Docker CE)	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2022.09.22	1.63.1	20.10.17 (Docker CE)	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2022.09.14	1.62.2	20.10.17 (Docker CE)	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2022.08.15	1.62.1	20.10.9	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2022.07.13	1.61.3	20.10.9	Privé

AMI Windows Server 2022 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2022-English-Full-ECS_Optimized-2022.06.15	1.61.2	20.10.9	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2022.01.18	1.57.1	20.10.9	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2021.12.16	1.57.1	20.10.7	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2021.11.11	1.57.0	20.10.7	Privé
Windows_Server-2022-English-Full-ECS_Optimized-2021.009.23	1.55.3	20.10.7	Privé

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-2024.02.13	`1.81.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2_English-Core-ECS_Optimized-2024.01.09	`1.79.2`	`20.10.23 (Docker CE)`	Public
Windows_Server-2_English-Core-ECS_Optimized-2023.12.12	`1.79.1`	`20.10.23 (Docker CE)`	Public
Windows_Server-2_English-Core-ECS_Optimized-2023.11.14	`1.79.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2022-English-Core-ECS_Optimized-2023.10.11	`1.77.0`	`20.10.21 (Docker CE)`	Public
Windows_Server-2022-English-Core-ECS_Optimized-2023.09.15	`1.75.3`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2022-English-Core-ECS	`1.74.1`	`20.10.21 (Docker CE)`	Privé

AMI de Windows Server 2022 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
_Optimized-2023.08.09			
Windows_Server-2022-English-Core-ECS_Optimized-2023.07.11	1.73.1	20.10.21 (Docker CE)	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2023.06.13	1.72.0	20.10.21 (Docker CE)	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2023.05.18	1.71.1	20.10.21 (Docker CE)	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2023.04.18	1.70.2	20.10.21 (Docker CE)	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2023.03.21	1.69.0	20.10.21 (Docker CE)	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2023.02.21	1.68.2	20.10.21 (Docker CE)	Privé

AMI de Windows Server 2022 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2022-English-Core-ECS_Optimized-2023.01.11	`1.68.0`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2022.12.14	`1.67.2`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2022.11.09	`1.65.1`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2022.10.12	`1.64.0`	`20.10.17 (Docker CE)`	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2022.09.22	`1.63.1`	`20.10.17 (Docker CE)`	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2022.09.14	`1.62.2`	`20.10.17 (Docker CE)`	Privé

AMI de Windows Server 2022 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2022-English-Core-ECS_Optimized-2022.08.15	1.62.1	20.10.9	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2022.07.13	1.61.3	20.10.9	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2022.06.15	1.61.2	20.10.9	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2021.12.16	1.57.1	20.10.7	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2021.11.11	1.57.0	20.10.7	Privé
Windows_Server-2022-English-Core-ECS_Optimized-2021.009.23	1.55.3	20.10.7	Privé

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-2024.02.13	`1.81.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2019-Englis-Full-ECS_Optimized-2024.01.09	`1.79.2`	`20.10.23 (Docker CE)`	Public
Windows_Server-2019-Englis-Full-ECS_Optimized-2023.12.12	`1.79.1`	`20.10.23 (Docker CE)`	Public
Windows_Server-2019-Englis-Full-ECS_Optimized-2023.11.14	`1.79.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2019-English-Full-ECS_Optimized-2023.10.11	`1.77.0`	`20.10.21 (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-English-Full-ECS_Optimized-2023.09.15	1.75.3	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2023.08.09	1.74.1	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2023.07.11	1.73.1	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2023.06.13	1.72.0	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2023.05.18	1.71.1	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2023.04.18	1.70.2	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2023.03.21	1.69.0	20.10.21 (Docker CE)	Privé

AMI Windows Server 2019 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Full-ECS_Optimized-2023.02.21	1.68.2	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2023.01.11	1.68.0	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2022.12.14	1.67.2	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2022.11.09	1.65.1	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2022.10.12	1.64.0	20.10.17 (Docker CE)	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2022.09.22	1.63.1	20.10.17 (Docker CE)	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2022.09.14	1.62.2	20.10.17 (Docker CE)	Privé

AMI Windows Server 2019 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Full-ECS_Optimized-2022.08.15	1.62.1	20.10.9	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2022.07.13	1.61.3	20.10.9	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2022.06.15	1.61.2	20.10.9	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2022.01.18	1.57.1	20.10.9	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2021.12.16	1.57.1	20.10.7	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2021.11.11	1.57.0	20.10.7	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2021.09.23	1.55.3	20.10.7	Privé

AMI Windows Server 2019 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Full-ECS_Optimized-2021.08.12	1.55.0	20.10.6	Public
Windows_Server-2019-English-Full-ECS_Optimized-2021.07.13	1.54.02	20.10.6	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2021.07.08	1.54.0	20.10.5	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2021.06.11	1.53.0	20.10.5	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2021.05.21	1.52.2	20.10.4	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2021.04.14	1.51.0	20.10.0	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2021.03.11	1.50.2	19.03.14	Privé

AMI Windows Server 2019 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Full-ECS_Optimized-2021.02.10	1.50.0	19.03.14	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2021.01.13	1.49.0	19.03.14	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2020.11.18	1.48.0	19.03.13	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2020.11.06	1.47.0	19.03.11	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2020.10.14	1.45.0	19.03.11	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2020.08.12	1.43.0	19.03.11	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2020.07.15	1.41.1	19.03.5	Privé

AMI Windows Server 2019 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Full-ECS_Optimized-2020.06.11	1.40.0	19.03.5	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2020.05.14	1.39.0	19.03.5	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2020.01.15	1.35.0	19.03.5	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2019.12.16	1.34.0	19.03.5	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2019.11.25	1.34.0	19.03.4	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2019.11.13	1.32.1	19.03.4	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2019.10.09	1.32.0	19.03.2	Privé

AMI Windows Server 2019 Full optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Full-ECS_Optimized-2019.09.11	1.30.0	19.03.1	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2019.08.16	1.29.1	19.03.1	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2019.07.19	1.29.0	18.09.8	Privé
Windows_Server-2019-English-Full-ECS_Optimized-2019.05.10	1.27.0	18.09.4	Privé

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-2024.02.13	`1.81.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2019-Englis-Core-ECS_Optimized-2024.01.09	`1.79.2`	`20.10.23 (Docker CE)`	Public
Windows_Server-2019-Englis-Core-ECS_Optimized-2023.12.12	`1.79.1`	`20.10.23 (Docker CE)`	Public
Windows_Server-2019-Englis-Core-ECS_Optimized-2023.11.14	`1.79.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2019-English-Core-ECS_Optimized-2023.10.11	`1.77.0`	`20.10.21 (Docker CE)`	Public
Windows_Server-2019-English-Core-ECS_Optimized-2023.09.15	`1.75.3`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2019-English-Core-ECS	`1.74.1`	`20.10.21 (Docker CE)`	Privé

AMI de Windows Server 2019 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
_Optimized-2023.08.09			
Windows_Server-2019-English-Core-ECS_Optimized-2023.07.11	1.73.1	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2023.06.13	1.72.0	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2023.05.18	1.71.1	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2023.04.18	1.70.2	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2023.03.21	1.69.0	20.10.21 (Docker CE)	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2023.02.21	1.68.2	20.10.21 (Docker CE)	Privé

AMI de Windows Server 2019 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Core-ECS_Optimized-2023.01.11	`1.68.0`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2022.12.14	`1.67.2`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2022.11.09	`1.65.1`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2022.10.12	`1.64.0`	`20.10.17 (Docker CE)`	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2022.09.22	`1.63.1`	`20.10.17 (Docker CE)`	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2022.09.14	`1.62.2`	`20.10.17 (Docker CE)`	Privé

AMI de Windows Server 2019 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Core-ECS_Optimized-2022.08.15	1.62.1	20.10.9	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2022.07.13	1.61.3	20.10.9	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2022.06.15	1.61.2	20.10.9	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2022.01.18	1.57.1	20.10.9	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2021.12.16	1.57.1	20.10.7	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2021.11.11	1.57.0	20.10.7	Privé

AMI de Windows Server 2019 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Core-ECS_Optimized-2021.09.23	1.55.3	20.10.7	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2021.08.12	1.55.0	20.10.6	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2021.07.13	1.54.02	20.10.6	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2021.07.08	1.54.0	20.10.6	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2021.06.11	1.53.0	20.10.5	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2021.05.21	1.52.2	20.10.4	Privé

AMI de Windows Server 2019 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Core-ECS_Optimized-2021.04.14	1.51.0	20.10.0	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2021.03.11	1.50.2	19.03.14	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2021.02.10	1.50.0	19.03.14	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2021.01.13	1.49.0	19.03.14	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2020.11.18	1.48.0	19.03.13	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2020.11.06	1.47.0	19.03.11	Privé

AMI de Windows Server 2019 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Core-ECS_Optimized-2020.10.14	1.45.0	19.03.11	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2020.09.09	1.44.3	19.03.11	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2020.08.12	1.43.0	19.03.11	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2020.07.15	1.41.1	19.03.5	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2020.06.11	1.40.0	19.03.5	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2020.05.14	1.39.0	19.03.5	Privé

AMI de Windows Server 2019 Core optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Core-ECS_Optimized-2020.01.15	`1.35.0`	`19.03.5`	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2019.12.16	`1.34.0`	`19.03.5`	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2019.11.25	`1.34.0`	`19.03.4`	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2019.11.13	`1.32.1`	`19.03.4`	Privé
Windows_Server-2019-English-Core-ECS_Optimized-2019.10.09	`1.32.0`	`19.03.2`	Privé

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

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-2024.02.13	`1.81.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS_Optimized-2024.01.09	`1.79.2`	`20.10.23 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS_Optimized-2023.12.12	`1.79.1`	`20.10.23 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS_Optimized-2023.11.14	`1.79.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS_Optimized-2023.10.11	`1.77.0`	`20.10.21 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS	`1.75.3`	`20.10.21 (Docker CE)`	Privé

AMI complète de Windows Server 2016 optimisée pour Amazon ECS	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
_Optimized-2023.09.15			
Windows_Server-2016-English-Full-ECS_Optimized-2023.08.09	`1.74.1`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2023.07.11	`1.73.1`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2023.06.13	`1.72.0`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2023.05.18	`1.71.1`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2023.04.18	`1.70.2`	`20.10.21 (Docker CE)`	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2023.03.21	`1.69.0`	`20.10.21 (Docker CE)`	Privé

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-2023.02.21	1.68.2	20.10.21 (Docker CE)	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2023.01.11	1.68.0	20.10.21 (Docker CE)	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2022.12.14	1.67.2	20.10.21 (Docker CE)	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2022.11.09	1.65.1	20.10.21 (Docker CE)	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2022.10.12	1.64.0	20.10.17 (Docker CE)	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2022.09.22	1.63.1	20.10.17 (Docker CE)	Privé

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-2022.09.14	1.62.2	20.10.17 (Docker CE)	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2022.08.15	1.62.1	20.10.9	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2022.07.13	1.61.3	20.10.9	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2022.06.15	1.61.2	20.10.9	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2022.01.18	1.57.1	20.10.9	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2021.12.16	1.57.1	20.10.7	Privé

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-2021.11.11	1.57.0	20.10.7	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2021.09.23	1.55.3	20.10.7	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2021.08.12	1.55.0	20.10.6	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2021.07.13	1.54.02	20.10.6	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2021.07.08	1.54.0	20.10.5	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2021.06.11	1.53.0	20.10.5	Privé

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-2021.05.21	1.52.2	20.10.4	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2021.04.14	1.51.0	20.10.0	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2021.03.11	1.50.2	19.03.14	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2021.02.10	1.50.0	19.03.14	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2021.01.13	1.49.0	19.03.14	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2020.11.18	1.48.0	19.03.13	Privé

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-2020.11.06	1.47.0	19.03.11	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2020.10.14	1.45.0	19.03.12	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2020.09.09	1.44.3	19.03.11	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2020.08.12	1.43.0	19.03.11	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2020.07.15	1.41.1	19.03.5	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2020.06.11	1.40.0	19.03.5	Privé

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-2020.05.14	1.39.0	19.03.5	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2020.01.15	1.35.0	19.03.5	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2019.12.16	1.34.0	19.03.5	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2019.11.25	1.34.0	19.03.4	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2019.11.13	1.32.1	19.03.4	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2019.10.09	1.32.0	19.03.2	Privé

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-2019.09.11	1.30.0	19.03.1	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2019.08.16	1.29.1	19.03.1	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2019.07.19	1.29.0	18.09.8	Privé
Windows_Server-2016-English-Full-ECS_Optimized-2019.03.07	1.26.0	18.03.1	Privé

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

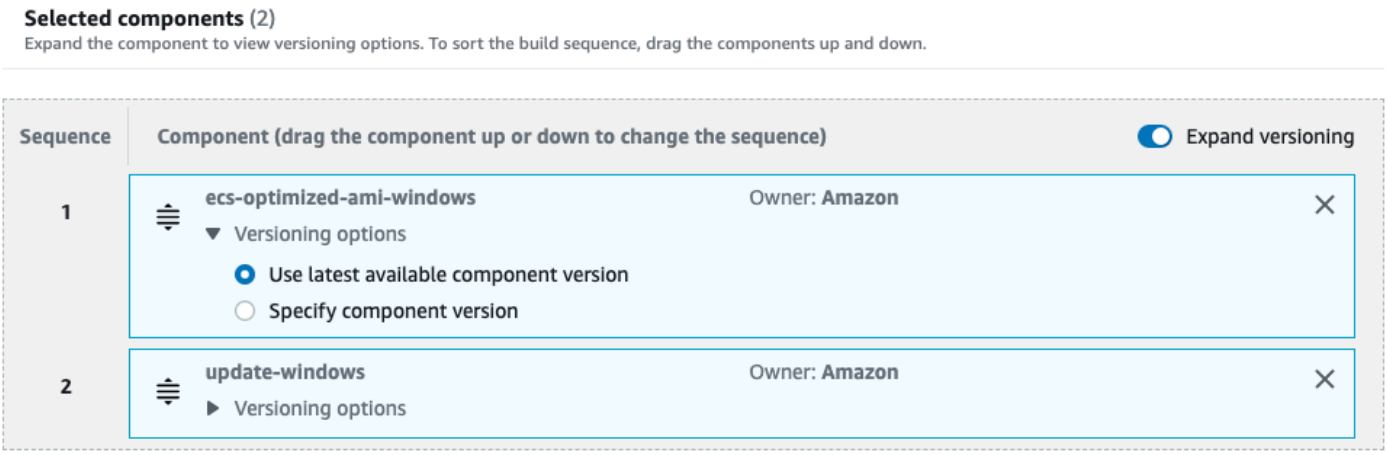
Utilisez EC2 Image Builder pour créer votre propre AMI Windows personnalisée optimisée pour Amazon ECS. Cela facilite l'utilisation d'une AMI Windows avec votre propre licence sur Amazon ECS. Amazon ECS propose un composant Image Builder géré qui fournit la configuration système nécessaire pour exécuter 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 démonstration complète de l'utilisation d'EC2 Image Builder, veuillez consulter Getting started with EC2 Image Builder dans le Guide de l'utilisateur 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 2016 Full, 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 plus d'informations, consultez Établissement de la liste des versions de composants `ecs-optimized-ami-windows`.

Établissement de la liste des versions de composants **`ecs-optimized-ami-windows`**

Lors de la création d'une recette EC2 Image Builder et de la spécification du composant `ecs-optimized-ami-windows`, vous pouvez utiliser l'option par défaut ou 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.

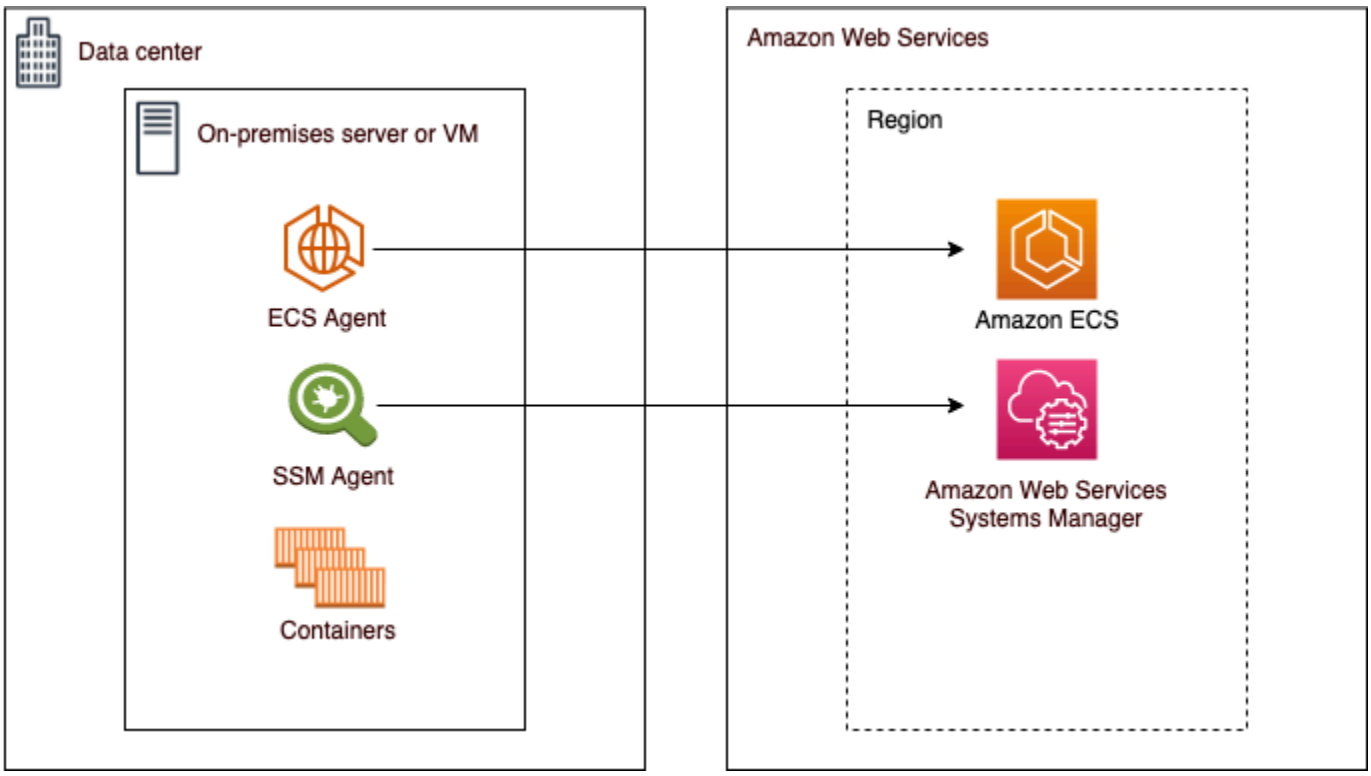
Pour répertorier les versions de composants **`ecs-optimized-ami-windows`** disponibles

1. Ouvrez la console EC2 Image Builder sur 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.

Instances externes (Amazon ECS Anywhere)

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.



Rubriques

- Systèmes d'exploitation et architectures système pris en charge
- Considérations
- Enregistrement d'une instance externe dans un cluster
- Annuler l'enregistrement d'une instance externe
- Mise à jour de l' AWS Systems Manager agent et de l'agent de conteneur Amazon ECS sur une instance externe

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
- CentOS 7
- CentOS Stream 8

- RHEL 7, RHEL 8 — Ni les référentiels de paquets ouverts de Docker ou 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
- openSUSE Tumbleweed
- Ubuntu 18, Ubuntu 20, Ubuntu 22
- 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.

- 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 Annuler l'enregistrement d'une instance externe.
- Vos instances externes ont besoin d'un rôle IAM qui leur permet de communiquer avec les AWS API. Pour plus d’informations, consultez Rôle IAM 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 plus d'informations, consultez Attributs personnalisés.
- Vous pouvez ajouter des balises de ressources à votre instance externe. Pour plus d'informations, consultez Marquage d'une instance de conteneur externe.
- ECS Exec est pris en charge sur les instances externes. Pour plus d'informations, consultez Surveillez les conteneurs Amazon ECS avec ECS Exec.
- Vous trouverez ci-dessous des considérations supplémentaires spécifiques aux réseaux avec vos instances externes. Pour plus d'informations, consultez Réseaux avec ECS Anywhere.
  - 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 plus d'informations, consultez Utilisation des rôles IAM de tâche sur vos instances Amazon EC2 ou externes.

```
$ 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 avec ECS Anywhere

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 Choix d'un mode réseaudans le Guide des bonnes pratiques Amazon ECS.
- 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 (région)* représente l'identifiant de région d'une région AWS prise en charge par Amazon ECS, telle que `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 “Utilisation des GMSA pour les conteneurs Windows sur Amazon EC2”.
- 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.

Enregistrement d'une instance externe dans un cluster

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 plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

AWS Management Console

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 page Cluster : ***name*** (Cluster : nom), 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 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 plus d'informations, consultez Rôle IAM 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 sur GNUpg, consultez le site Web 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 de plusieurs GPU dans votre définition de tâche.

```
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 plus d'informations, consultez Rôle IAM 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.

Annuler l'enregistrement d'une instance externe

Nous vous recommandons, une fois que vous avez fini d'utiliser une instance externe, d'annuler l'enregistrement de l'instance d'Amazon ECS et de AWS Systems Manager. 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.

Pour enregistrer une instance externe dans un nouveau cluster, une fois que l'instance externe a été désenregistrée d'Amazon ECS et de Systems Manager, vous pouvez nettoyer les AWS ressources restantes de l'instance et l'enregistrer auprès d'un nouveau cluster.

AWS Management Console

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 page Cluster : ***name*** (Cluster : nom), 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 page Container Instance : ***id*** (Instance de conteneur : id), choisissez Deregister (Annuler l'enregistrement).
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 Annuler l'enregistrement des instances gérées 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

Système d’exploitation	Étapes	
	sudo yum remove -y amazon-ecs-init amazon-ssm-agent Pour SUSE Enterprise Server 15 **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**	

Système d'exploitation	Étapes

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.	

Système d’exploitation	Étapes	

	`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`	

Système d’exploitation	Étapes	

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`	

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) 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) RHEL 7 (aarch64)	`amazon-ecs-init-latest.aarch64.rpm`
Debian 9 (x86_64)	`amazon-ecs-init-latest.amd64.deb`

Systèmes d'exploitation (architecture)	package ecs-init
Debian 10 (x86_64) Ubuntu 18 (x86_64) Ubuntu 20 (x86_64)	
Debian 9 (aarch64) Debian 10 (aarch64) Ubuntu 18 (aarch64) Ubuntu 20 (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'identifiant *`<region>`* dans la commande par le nom de la région (par exemple, `us-west-2`) dont vous êtes géographiquement le plus proche.

   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 sur GNUpg, consultez le site Web 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'identifiant *<region>* dans la commande par le nom de la région (par exemple, `us-west-2`) dont vous êtes géographiquement le plus proche.

   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
```

Gestion de capacité

AWS gère votre capacité Fargate, mais vous êtes responsable de la gestion de vos instances EC2 et de vos instances externes.

La gestion inclut l'application de correctifs de sécurité, ainsi que la mise à jour des instances de conteneur, afin que vous ayez accès aux dernières fonctionnalités d'Amazon ECS.

Outre la gestion des agents de conteneur, une gestion d'instance supplémentaire peut être requise en fonction de votre configuration. Cela consiste notamment à :

- Lancement d'une instance de conteneur
- Amorçage d'une instance de conteneur
- Démarrage d'une tâche au moment du lancement
- Utilisation de jonction d'ENI
- Gestion de la mémoire
- Gestion à distance de votre instance de conteneur
- Utilisation d'un proxy HTTP pour l'agent de conteneur et le démon Docker
- Mise à jour de l'agent de conteneur
- Annulation de l'enregistrement d'une instance de conteneur
- Drainage d'instance géré

Rubriques

- Agent de conteneur Amazon ECS Linux

- Gestion des instances de conteneur Linux
- Gestion des instances de conteneur Windows
- Drainage d'instance géré par Amazon ECS

Pour plus d'informations sur les fournisseurs de capacité Fargate et les fournisseurs de capacité du groupe Amazon EC2 Auto Scaling, consultez la section Fournisseurs de capacités.

Agent de conteneur Amazon ECS Linux

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 connaître les fonctionnalités et les améliorations 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 `1.9.0`.

Cycle de vie

Lorsque l'agent de conteneur Amazon ECS enregistre une instance Amazon EC2 dans votre cluster, le statut de l'instance Amazon EC2 est `ACTIVE` et le statut de connexion de l'agent est `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 des ID d'instance de conteneur en double pour la même instance Amazon EC2, vous pouvez annuler en toute sécurité l'enregistrement des doublons qui sont répertoriés avec le statut `ACTIVE` et avec le statut de connexion d'agent `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 plus d'informations, consultez Drainage des instances de conteneur.

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`. Pour plus d'informations sur les hooks de cycle de vie Auto Scaling, consultez Hooks de cycle de vie Amazon EC2 Auto Scaling.

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 plus d'informations, consultez

Extraction des métadonnées d'AMI optimisée 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.

Utilisation d'autres systèmes d'exploitation Linux

Pour installer la dernière version de l'agent de conteneur Amazon ECS sur un autre système d'exploitation, veuillez consulter Installation de l'agent de conteneur Amazon ECS. Le tableau fourni dans AMI optimisée pour Amazon ECS montre la version de Docker qui est testée sur Amazon Linux 2 pour chaque version de l'agent.

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 les anciens emballages, consultez le journal des modifications d'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

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. Les variables d'environnement suivantes sont disponibles, et sont toutes facultatives.

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 utilisateur Amazon EC2 lors du lancement. Pour plus d'informations, consultez Démarrage des instances de conteneur avec les données utilisateur d'Amazon EC2.

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 au lancement sur votre instance Windows dans le Guide de l'utilisateur Amazon EC2 pour les instances WIndows et the section called “Démarrage des instances de conteneur”.

Si vous lancez manuellement l'agent de conteneur Amazon ECS (pour les AMI non optimisées pour Amazon ECS), vous pouvez utiliser ces variables d'environnement dans la commande docker run qui vous permet de 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
```

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 d'instance de conteneur dans Simple Storage Service (Amazon S3)

La configuration de l'agent de conteneur Amazon ECS est contrôlée avec les variables d'environnement décrites à la section précédente. 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. Certaines variables d'environnement inoffensives, telles que `ECS_CLUSTER`, peuvent être transmises à l'instance de conteneur au moment du lancement via les données utilisateur Amazon EC2 et enregistrées 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 utilisateur Amazon EC2 pour installer AWS CLI et copier vos informations de configuration au `/etc/ecs/ecs.config` moment du lancement de l'instance.

Pour permettre à votre rôle d'instance de conteneur d'accéder en lecture seule à Amazon S3

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Roles (Rôles), puis sélectionnez le rôle IAM à utiliser pour vos instances de conteneur. Ce rôle est probablement intitulé `ecsInstanceRole`. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.
3. Sous Managed Policies (Politiques gérées), choisissez Attach Policy (Attacher la politique).
4. Pour affiner les résultats de la politique , sur la page Attach Policy (Attacher la politique), dans le champ Filtre, tapez `S3`.
5. Cochez la case située à gauche de la politique d'AmazonS3 et choisissez Joindre la ReadOnlyAccess politique.

Pour stocker un fichier **`ecs.config`** dans Amazon S3

1. 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 plus d'informations, consultez Authentification de registre privé pour les tâches.

   ```
   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.

2. Pour stocker votre fichier de configuration, créez un compartiment privé dans Amazon S3. Pour plus d'informations, consultez Création d'un compartiment dans le Guide de l'utilisateur d'Amazon Simple Storage Service.

3. Chargez le fichier `ecs.config` dans votre compartiment S3. Pour plus d'informations, consultez Ajouter un objet à un compartiment 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 en suivant les étapes de Lancement d'une instance de conteneur Amazon ECS Linux. DansStep 6.g, utilisez l'exemple de script suivant qui installe AWS CLI et copie votre fichier de configuration dans. `/etc/ecs/ecs.config`

```
#!/bin/bash
yum install -y aws-cli
aws s3 cp s3://your_bucket_name/ecs.config /etc/ecs/ecs.config
```

Authentification de registre privé pour les instances de conteneur

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 EC2 ou Fargate. Pour plus d'informations, consultez Authentification de registre privé pour les tâches.

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 fichier `/etc/ecs/ecs.config` pour ces variables lorsque l'instance de conteneur démarre et chaque fois que le

service est démarré (avec la commande sudo start ecs). Les AMI qui ne sont pas optimisées par Amazon ECS doivent stocker ces variables d'environnement dans un fichier et les transmettre avec l'option `--env-file` *`path_to_env_file`* à la commande docker run qui démarre l'agent de conteneur.

> **Important**
>
> Nous déconseillons d'injecter ces variables d'environnement d'authentification au moment du lancement de l'instance avec les données utilisateur Amazon EC2 ou de les transmettre avec l'option `--env` à la commande docker run. 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 d'instance de conteneur dans Simple Storage Service (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"}}
```

Activation des registres privés

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 plus d'informations, consultez 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 plus d'informations, consultez the section called "Introspection des conteneurs".

```
curl http://localhost:51678/v1/metadata
```

Tâche automatisée et nettoyage d'image

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 plus d'informations, consultez 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.

Paramètres ajustables

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`

Cette variable spécifie le temps d'attente avant la suppression des conteneurs qui appartiennent à des tâches arrêtées. Le processus de nettoyage d'image ne peut pas supprimer une image tant qu'un conteneur y fait référence. Une fois que les images ne sont plus référencées par des conteneurs (arrêtés ou en cours d'exécution), l'image peut être nettoyée. Par défaut, ce paramètre est défini sur trois heures, mais vous pouvez réduire cette période à une seconde, si votre application le nécessite. Le paramètre est ignoré si vous définissez la valeur sur moins d'une seconde.

`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.

Flux de travail de nettoyage

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.

Ajouter un attribut à une instance de conteneur Amazon ECS

Vous pouvez ajouter des attributs personnalisés lors de l'enregistrement des instances à l'aide de l'agent de conteneur ou manuellement, en utilisant la AWS Management Console. Pour plus d'informations sur les paramètres de configuration de l'agent de conteneur Amazon ECS disponibles, consultez Amazon ECS Container Agent on GitHub.

Pour ajouter des attributs personnalisés à l'aide de la console

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters, puis choisissez le cluster.
3. Choisissez Infrastructure.
4. Choisissez l'instance de conteneur, puis sélectionnez Attributs.
5. Choisissez Ajouter un attribut personnalisé.
6. Pour Nom de l'attribut et Valeur, saisissez un nom et une valeur pour l'attribut, puis choisissez Envoyer.

Répétez cette procédure pour chaque attribut que vous voulez ajouter.

Ajout d'attributs personnalisés à l'aide du AWS CLI

Les exemples suivants montrent comment ajouter des attributs personnalisés à l'aide de la commande put-attributes.

Exemple : un seul attribut

L'exemple suivant ajoute l'attribut personnalisé « stack=prod » à l'instance de conteneur spécifiée dans le cluster par défaut.

```
aws ecs put-attributes --attributes name=stack,value=prod,targetId=arn
```

Exemple : plusieurs attributs

L'exemple suivant ajoute les attributs personnalisés « stack=prod » et « project=a » à l'instance de conteneur spécifiée dans le cluster par défaut.

```
aws ecs put-attributes --attributes name=stack,value=prod,targetId=arn
 name=project,value=a,targetId=arn
```

Filtrage par attribut à l'aide de la console

Vous pouvez appliquer un filtre à vos instances de conteneur, ce qui vous permet de voir les attributs personnalisés.

Filtrer les instances de conteneur par attribut à l'aide de la console

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters, puis choisissez le cluster.
3. Choisissez Infrastructure.
4. Choisissez l'instance de conteneur.
5. Dans le champ de texte Filtrer les attributs personnalisés par clé ou par valeur, saisissez les attributs selon lesquels le filtre sera appliqué. Le format doit être AttributeName: AttributeValue.

Filtrez les instances de conteneur par attribut à l'aide du AWS CLI

Les exemples suivants montrent comment filtrer les instances de conteneur par attribut à l'aide de la list-constainer-instancescommande. Pour en savoir plus sur la syntaxe des filtres, consultez Création d'expressions pour définir des instances de conteneur pour les tâches Amazon ECS.

Exemple : attribut intégré

L'exemple suivant utilise des attributs intégrés afin de répertorier les instances g2.2xlarge.

```
aws ecs list-container-instances --filter "attribute:ecs.instance-type == g2.2xlarge"
```

Exemple : attribut personnalisé

L'exemple suivant répertorie les instances avec l'attribut personnalisé « stack=prod ».

```
aws ecs list-container-instances --filter "attribute:stack == prod"
```

Exemple : exclure une valeur d'attribut

L'exemple suivant répertorie les instances avec l'attribut personnalisé « stack » à moins que la valeur de l'attribut ne soit « prod ».

```
aws ecs list-container-instances --filter "attribute:stack != prod"
```

Exemple : plusieurs valeurs d'attributs

L'exemple suivant utilise des attributs intégrés afin de répertorier les instances de type `t2.small` ou `t2.medium`.

```
aws ecs list-container-instances --filter "attribute:ecs.instance-type in [t2.small,
 t2.medium]"
```

Exemple : plusieurs attributs

L'exemple suivant utilise des attributs intégrés afin de répertorier les instances T2 dans la zone de disponibilité us-east-1a.

```
aws ecs list-container-instances --filter "attribute:ecs.instance-type =~ t2.* and
 attribute:ecs.availability-zone == us-east-1a"
```

Gestion des instances de conteneur Linux

La gestion des instances de conteneur Linux comprend :

- Lancement d'une instance de conteneur
- Amorçage d'une instance de conteneur
- Démarrage d'une tâche au moment du lancement
- Utilisation de jonction d'ENI
- Gestion de la mémoire
- Gestion à distance de votre instance de conteneur
- Utilisation d'un proxy HTTP pour l'agent de conteneur et le démon Docker
- Mise à jour de l'agent de conteneur

  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 connaître les fonctionnalités et les améliorations 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 `1.9.0`.

Lancement d'une instance de conteneur Amazon ECS Linux

Vos instances de conteneur Amazon ECS sont créées à l'aide de la console Amazon EC2. Avant de commencer, assurez-vous d’avoir terminé les étapes de Configurer l'utilisation d'Amazon ECS.

Vous pouvez lancer une instance par différentes méthodes, notamment la console Amazon EC2 et AWS CLI le SDK. La procédure de cette page couvre l'assistant de lancement de la console Amazon EC2. Pour plus d'informations sur les autres méthodes de lancement d'instance, consultez Lancer votre instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

Pour plus d'informations sur l'assistant de lancement, consultez Lancement d'une instance à l'aide du nouvel assistant de lancement d'instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

Nouvel assistant de lancement d'instance Amazon EC2

Vous pouvez utiliser le nouvel assistant Amazon EC2 pour lancer une instance. L'assistant de lancement d'instance spécifie les paramètres de lancement qui sont requis 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.

Paramètres pour la configuration d'instance

- Commencer le lancement de l'instance
- 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

Commencer le lancement de l'instance

1. Ouvrez la console Amazon EC2 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. Sur le tableau de bord de la console Amazon EC2, sélectionnez Launch instance (Lancer une 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 les dernières AMI optimisées pour Amazon ECS et leurs valeurs, veuillez consulter AMI optimisée pour Amazon ECS Linux.
2. Appuyez sur Entrée.
3. Sur la page Sélection d'une Amazon Machine Image (AMI) sélectionnez l'onglet AMI 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 les Instances Linux.

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 Configuration du proxy HTTP pour les instances de conteneur 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 plus d'informations, consultez 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 utilisateur Amazon EC2 sont exécutés une seule fois, au 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 User data (Données utilisateur), en remplaçant *your_cluster_name* 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 remplaçant *your_bucket_name 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 d'instance de conteneur dans Simple Storage Service (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 permet de créer des balises associées uniquement à Amazon ECS, qui ne peuvent pas être répertoriées à l'aide de l'API Amazon EC2.

  > **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_CONTENER_INSTANCE_TAGS pour ajouter des balises. Cela est dû à la manière dont les balises sont ajoutées aux instances Amazon EC2 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 les balises pour votre instance de conteneur, puis utilisez le paramètre de configuration `ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM` pour les propager depuis 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 plus d’informations, consultez Démarrage des instances de conteneur avec les données utilisateur d'Amazon EC2.

Ancien assistant de lancement d'instance Amazon EC2

Pour lancer une instance de conteneur

1. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Sur le tableau de bord EC2, sélectionnez Launch instance (Lancer une instance).
4. Sur la page Choose an Amazon Machine Image (AMI) (Choisir une Amazon Machine Image (AMI)), effectuez les étapes suivantes :

   a. Choisissez AWS Marketplace.

   b. Choisissez une AMI pour votre instance de conteneur. Vous pouvez rechercher l'une des AMI optimisées pour Amazon ECS, par exemple l'AMI Amazon Linux 2 optimisée pour Amazon ECS. Si vous ne choisissez pas une AMI optimisée pour Amazon ECS, vous devez suivre les procédures décrites dans Installation de l'agent de conteneur Amazon ECS.

      Pour de plus amples informations sur les dernières AMI optimisées pour Amazon ECS, veuillez consulter AMI optimisée pour Amazon ECS.
5. Sur la page Choose an Instance Type (Choisir un type d'instance), vous pouvez sélectionner la configuration matérielle de votre instance. Le type d'instance `t2.micro` est sélectionné par défaut. Le type d'instance que vous sélectionnez détermine les ressources disponibles pour les tâches à exécuter.

   Lorsque vous avez terminé, sélectionnez Next: Configure Instance Details (Suivant : Configurer les détails d'instance).
6. Dans la page Configure Instance Details (Configurer les détails d'instance), complétez les étapes suivantes :

   a. Définissez le champ Number of instances (Nombre d'instances) en fonction du nombre d'instances de conteneur à ajouter à votre cluster.

   b. (Facultatif) Pour utiliser des instances Spot, pour Purchasing option (Option d'achat), cochez la case en regard de Request Spot Instances (Demander des instances 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.

c. Pour Network (Réseau), choisissez le VPC dans lequel lancer votre instance de conteneur.

d. Pour Subnet (Sous-réseau), choisissez un sous-réseau à utiliser ou conservez l'option par défaut pour choisir le sous-réseau par défaut dans une zone de disponibilité.

e. Définissez le champ Auto-assign Public IP (Attribuer automatiquement l'adresse IP publique) pour indiquer si vous souhaitez que votre instance soit accessible ou non à partir du réseau Internet public. 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 service. 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, veuillez consulter 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 Configuration du proxy HTTP pour les instances de conteneur Linux dans ce guide. Pour plus d'informations, consultez the section called “Créer un Virtual Private Cloud”.

f. Sélectionnez le rôle IAM de votre 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 plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.

g. (Facultatif) 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 utilisateur Amazon EC2 sont exécutés une seule fois, au 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 User data (Données utilisateur), en remplaçant *your_cluster_name* 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 remplaçant *your_bucket_name 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 d'instance de conteneur dans Simple Storage Service (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 permet de créer des balises

associées uniquement à Amazon ECS, qui ne peuvent pas être répertoriées à l'aide de l'API Amazon EC2.

> **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_CONTENER_INSTANCE_TAGS pour ajouter des balises. Cela est dû à la manière dont les balises sont ajoutées aux instances Amazon EC2 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 les balises pour votre instance de conteneur, puis utilisez le paramètre de configuration `ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM` pour les propager depuis 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 plus d'informations, consultez Démarrage des instances de conteneur avec les données utilisateur d'Amazon EC2.

h. Choisissez Next: Add Storage (Suivant : Ajouter le stockage).

7. Sur la page Add Storage (Ajouter le 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.

Lorsque vous avez terminé de configurer vos volumes, choisissez Next: Add Tags (Suivant : Ajouter des étiquettes).

8. Sur la page Add Tags (Ajouter des étiquettes), spécifiez les étiquettes en fournissant les combinaisons clé et valeur pour l'instance de conteneur. Choisissez Add another tag (Ajouter une autre étiquette) pour ajouter plusieurs étiquettes à votre instance de conteneur. Pour plus d'informations sur les étiquettes de ressource, consultez Balisage des ressources Amazon ECS.

   Choisissez Next: Configure Security Group (Suivant : Configurer le groupe de sécurité) une fois que vous avez terminé.

9. Sur la page Configure Security Group (Configurer le groupe 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. Sélectionnez ou créez un groupe de sécurité de la façon suivante, puis choisissez Review and Launch (Vérifier et lancer).

10. Sur la page Review Instance Launch (Examiner le lancement de l'instance), sous Security Groups (Groupes de sécurité), vous verrez que l'assistant a créé et sélectionné un groupe de sécurité pour vous. Au lieu de cela, sélectionnez le groupe de sécurité que vous avez créé à l'étape Configurer l'utilisation d'Amazon ECS en procédant comme suit :

    a. Sélectionnez Edit security groups (Modifier les groupes de sécurité).

    b. Sur la page Configure Security Group (Configurer le groupe de sécurité), choisissez l'option Select an existing security group (Sélectionnez un groupe de sécurité existant).

    c. Sélectionnez le groupe de sécurité que vous avez créé pour votre instance de conteneur dans la liste des groupes de sécurité existants, puis choisissez Review and Launch (Vérifier et lancer).

11. Sur la page Review Instance Launch (Examiner le lancement d'instance), sélectionnez Launch (Lancer).
12. Dans la boîte de dialogue Sélectionnez une paire de clés existante ou créez une nouvelle paire de clés, sélectionnez Choisir une paire de clés existante, puis sélectionnez la paire de clés créée lors de la configuration.

    Lorsque vous êtes prêt, cochez la case de confirmation, puis sélectionnez Launch Instances (Lancer des instances).
13. 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.
14. Sur l'écran Instances, vous pouvez afficher le statut de votre instance. Il suffit de peu de temps pour lancer une instance. Lorsque vous lancez une instance, son état initial est `pending`. Une fois que l'instance a démarré, son état devient `running` et elle reçoit un nom DNS public. Si la colonne DNS public est masquée, choisissez l'icône Afficher/Masquer, DNS public.

Utilisation des instances Spot

Une instance Spot est une instance Amazon EC2 non utilisée qui est disponible à un prix inférieur au prix à la demande. Comme une instance Spot vous permet de demander des instances EC2 inutilisées avec de fortes remises, vous pouvez réduire considérablement vos coûts Amazon EC2. Le prix horaire d'une instance Spot est appelé prix spot. Le prix Spot de chaque type d'instance dans chaque zone de disponibilité est défini par Amazon EC2 et varie en fonction de l'offre et de la demande à long terme pour les Instances Spot. Pour plus d'informations, veuillez consulter Instances Spot dans le Guide de l'utilisateur Amazon EC2 pour instances Linux.

Vous pouvez enregistrer des instances Spot auprès de vos clusters Amazon ECS. Pour plus d'informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

Drainage des instances Spot

Amazon EC2 résilie, arrête ou met en veille prolongée votre instance Spot lorsque le prix Spot dépasse le prix maximum de votre demande ou lorsque la capacité n'est plus disponible. Amazon EC2 fournit un avis d'interruption de deux minutes pour les actions de résiliation et d'arrêt des instances Spot. 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, ECS reçoit l'avis d'interruption d'instance Spot et bascule l'instance à l'état `DRAINING`.

> **Important**
>
> Amazon ECS ne reçoit pas d'avis de la part d'Amazon EC2 lorsque les instances sont supprimées par le rééquilibrage de la capacité Auto Scaling. Pour plus d'informations, consultez Rééquilibrage de la capacité Amazon EC2 Auto Scaling.

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 et doit être activé manuellement. Pour activer le drainage d'une instance Spot pour une nouvelle instance de conteneur, lors du lancement de l'instance de conteneur, ajoutez le script suivant dans le champ Données utilisateur, en le *`MyCluster`*remplaçant 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 plus d'informations, consultez 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
     ```
   - Pour l'AMI Amazon Linux optimisée pour Amazon ECS :

```
sudo stop ecs && sudo start 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 plus d'informations, consultez the section called “Introspection des conteneurs”.

   ```
   curl http://localhost:51678/v1/metadata
   ```

Drainage des instances de conteneur

Il se peut que vous deviez supprimer une instance de conteneur de votre cluster, par exemple pour effectuer des mises à jour du système, mettre à jour le démon Docker ou 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 Types de déploiement Amazon ECS et Paramètres de définition de service.

- 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.

Drainage des instances de conteneur

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez un cluster qui héberge vos instances.
4. Sur la page Cluster : ***name*** (Cluster : nom), 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).

Démarrage des instances de conteneur avec les données utilisateur d'Amazon EC2

Lorsque vous lancez une instance Amazon EC2, vous avez la possibilité de transmettre les données utilisateur à cette instance. 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 de cloud, des scripts shell et des directives `cloud-init`. 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 lors de l'utilisation de l'Amazon EC2 Launch Wizard. Pour plus d’informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

Rubriques

- Agent de conteneur Amazon ECS
- Démon 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 lors du lancement avec les données utilisateur Amazon EC2. 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 des informations de configuration du démon Docker avec les données utilisateur Amazon EC2. Pour plus d'informations sur les options de configuration , consultez la documentation sur le démon Docker.

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
```

Démarrage d'une tâche au moment du lancement d'une instance de conteneur

En fonction de la conception de l'architecture de votre application, vous devrez peut-être exécuter un conteneur spécifique sur chaque instance de conteneur pour traiter les problèmes de sécurité et de fonctionnement, telles que la surveillance, la sécurité, les métriques, la découverte de service 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. Ensuite, utilisez Amazon ECS pour placer la tâche au moment du lancement avec les données utilisateur Amazon EC2.

Le script de données utilisateur Amazon EC2 de 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. Si vous ne l'avez pas déjà fait, créez une définition de tâche avec le conteneur que vous souhaitez exécuter sur votre instance de conteneur au moment du lancement en suivant les procédures décrites dans Création d'une définition de tâche à l'aide de la console.
2. Modifiez votre rôle IAM `ecsInstanceRole` pour ajouter des autorisations pour l'opération d'API `StartTask`. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.
   a. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
   b. Dans le panneau de navigation, sélectionnez Rôles.
   c. Cliquez sur le bouton `ecsInstanceRole`. Si le rôle n'existe pas, utilisez la procédure de Rôle IAM d'instance de conteneur Amazon ECS pour créer le rôle et revenez ensuite à la procédure actuelle. Si le rôle existe, sélectionnez-le pour afficher les politiques attachées.
   d. Sous l'onglet Permissions (Autorisations), choisissez Add inline policy (Ajouter une politique en ligne).
   e. Pour Service, sélectionnez Choose a service (Choisir un service), Elastic Container Service.
   f. Pour Actions, tapez StartTaskdans le champ de recherche, puis sélectionnez StartTask.
   g. Pour Ressources, sélectionnez All resources (Toutes les ressources), puis choisissez Review policy (Examiner une politique).
   h. Sur la page Review policy (Examiner une politique), entrez le nom de votre politique, par exemple `ecs-start-task`, puis sélectionnez Create policy (Créer une politique).
3. Lancez une ou plusieurs instances de conteneur avec l'AMI Amazon Linux 2 optimisée pour Amazon ECS en suivant la procédure décrite dans Lancement d'une instance de conteneur Amazon ECS Linux, mais dans Step 6.g copiez et coller le script de données utilisateur MIME en plusieurs parties ci-dessous dans le champ User data (Données utilisateur). Remplacez *`your_cluster_name`* par le cluster auprès duquel l'instance de conteneur est enregistrée et *`my_task_def`* par la définition de tâche à exécuter sur l'instance lors du lancement.

> **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==--
```

4. Vérifiez que vos instances de conteneur sont lancées dans le cluster approprié et que vos tâches ont démarré.

   a. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.

   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.

Jonction d'interface réseau Elastic

> **Note**
>
> Cette fonction n'est pas disponible sur Fargate.

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 qui peuvent être attachées à des instances Amazon EC2 est limité par défaut et l'interface réseau principale est considérée comme l'une d'elles. Par exemple, par défaut, une instance `c5.large` peut avoir jusqu'à trois ENI qui lui sont attachées. L'interface réseau principale de l'instance est considérée comme une interface réseau. Vous pouvez donc attacher deux ENI supplémentaires à l'instance. Étant donné que 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'instances Amazon EC2 pris en charge. Lorsque vous utilisez ces types d'instances et activez le paramètre de compte `awsvpcTrunking`, des ENI supplémentaires sont disponibles sur les instances de conteneur nouvellement lancées. Cette configuration vous permet de placer plus de tâches en utilisant le mode réseau `awsvpc` sur chaque instance de conteneur. Pour plus d'informations sur le paramétrage du `awsvpcTrunking` compte, consultezAccès aux fonctionnalités d'Amazon ECS via les paramètres du compte.

Grâce à cette fonctionnalité, une `c5.large` instance `awsvpcTrunking` activée est portée à une ENI limite 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 plus d'informations, consultez Mise en réseau des tâches pour les tâches sur les instances Amazon EC2.

Considérations

Plusieurs éléments doivent être pris en compte lors de l'utilisation de 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 instances Amazon EC2 lancées après 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 DNS IPv4 basées sur les ressources doivent être désactivées dans les instances Amazon EC2. Pour désactiver cette option, assurez-vous que l'option Activer les requêtes DNS IPV4 (registre A) basées sur les ressources 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
```

- Les instances Amazon EC2 dans les sous-réseaux partagés ne sont pas prises en charge. Elles ne pourront pas s'enregistrer dans un cluster si elles sont utilisées.
- Vos tâches Amazon ECS doivent utiliser le mode réseau `awsvpc` et le type de lancement EC2. 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 Amazon ECS 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 plus d'informations, consultez 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.

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 plus d'informations, consultez 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'instance Amazon EC2 pris en charge, et les limites de l'instance seront augmentées. ENI Pour obtenir la liste des types d’instances, consultez Types d'instances Amazon EC2 pris en charge. 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 plus d’informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

> **Important**
>
> Les requêtes DNS IPv4 basées sur les ressources doivent être désactivées dans les instances Amazon EC2. Pour désactiver cette option, assurez-vous que l'option Activer les requêtes DNS IPV4 (registre A) basées sur les ressources 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
```

- Get-ECS (AttributeList)AWS Tools for Windows PowerShell

```
Get-ECSAttributeList -TargetType container-instance -AttributeName ecs.awsvpc-trunk-
id -Region us-east-1
```

Types d'instances Amazon EC2 pris en charge

L'exemple suivant montre les types d'instances Amazon EC2 pris en charge et le nombre de tâches qui utilisent le mode réseau `awsvpc` pouvant être lancées sur chaque type d'instance avant et après l'activation du paramètre de compte `awsvpcTrunking`. Pour les limites d'elastic network interface (ENI) pour chaque type d'instance, ajoutez-en une à la limite de tâches actuelle, car l'interface réseau principale est prise en compte dans la limite, et ajoutez-en deux à la nouvelle limite de tâches, car l'interface réseau principale et l'interface réseau principale comptent à nouveau comme limite.

> **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

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
m5a.8xlarge	7	60
m5a.12xlarge	7	60

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
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
m5d.metal	14	120
m5n.large	2	10

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
m5n.xlarge	3	20
m5n.2xlarge	3	40
m5n.4xlarge	7	60
m5n.8xlarge	7	60
m5n.12xlarge	7	60
m5n.16xlarge	14	120
m5zn.large	2	14
m5zn.xlarge	3	31
m5zn.2xlarge	3	64
m5zn.3xlarge	7	98
m5zn.6xlarge	7	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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
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
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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
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
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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
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	13	120
m6idn.metal	13	120
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	13	120
m6in.metal	13	120

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
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
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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
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
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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
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
mac2.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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c5.24xlarge	14	120
c5a.large	2	10
c5a.xlarge	3	20
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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c5d.12xlarge	7	60
c5d.18xlarge	14	120
c5d.24xlarge	14	120
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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c6g.12xlarge	7	60
c6g.16xlarge	14	120
c6g.metal	14	120
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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c6i.large	2	10
c6i.xlarge	3	20
c6i.2xlarge	3	40
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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c6in.large	2	10
c6in.xlarge	3	20
c6in.2xlarge	3	40
c6in.4xlarge	7	60
c6in.8xlarge	7	90
c6in.12xlarge	7	120
c6in.16xlarge	14	120
c6in.24xlarge	14	120
c6in.32xlarge	13	120
c6in.metal	13	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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c7a.48xlarge	14	120
c7a.metal-48xl	14	120
c7g.medium	1	4
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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
c7gn.medium	1	4
c7gn.large	2	10
c7gn.xlarge	3	20
c7gn.2xlarge	3	40
c7gn.4xlarge	7	60
c7gn.8xlarge	7	60
c7gn.12xlarge	7	60
c7gn.16xlarge	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

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	13	120
r6idn.metal	13	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	13	120
r6in.metal	13	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
u-3tb1.56xlarge	7	12
u-6tb1.56xlarge	14	12
u-18tb1.112xlarge	14	12
u-18tb1.metal	14	12
u-24tb1.112xlarge	14	12
u-24tb1.metal	14	12

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
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
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

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
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

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
i4g.16xlarge	14	120
i4i.xlarge	3	8
i4i.2xlarge	3	28
i4i.4xlarge	7	58

Type d'instance	Limite de tâches sans ENI jonction	Limite de tâches avec ENI trunking
i4i.8xlarge	7	118
i4i.12xlarge	7	118
i4i.16xlarge	14	248
i4i.24xlarge	14	118
i4i.32xlarge	14	498
i4i.metal	14	498
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
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
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
trn1.2xlarge	3	19
trn1.32xlarge	39	120
trn1n.32xlarge	79	242
vt1.3xlarge	3	40
vt1.6xlarge	7	60
vt1.24xlarge	14	120

Gestion de la mémoire des instances de conteneurs

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 plateforme et de la mémoire utilisée par le noyau du système, ce nombre est différent de la quantité de mémoire installée qui est annoncée pour les instances Amazon EC2. 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 appeléeECS_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 avec les données utilisateur d'Amazon EC2.

Si vous spécifiez 8192 Mo pour la tâche et qu'aucune de vos instances de conteneur ne dispose de 8192 Mo ou plus de mémoire disponible 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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 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 .

Rubriques

- Stratégie IAM Run Command

- Utilisation de la fonctionnalité Exécuter la commande

Stratégie IAM Run Command

Pour pouvoir envoyer des commandes à vos instances de conteneur avec Run Command, vous devez lier une stratégie IAM qui autorise `ecsInstanceRole` à avoir accès aux API Systems Manager. La procédure suivant décrit comment lier les stratégies gérées Systems Manager à votre rôle d'instance de conteneur afin que les instances lancées avec ce rôle puissent utiliser Run Command.

Pour attacher les stratégies Systems Manager à votre **`ecsInstanceRole`**

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Rôles.
3. Sélectionnez `ecsInstanceRole`. Si le rôle n'existe pas, suivez les procédures décrites dans Rôle IAM d'instance de conteneur Amazon ECS pour le créer.
4. Choisissez l'onglet Permissions (Autorisations).
5. Choisissez Attach Policies (Attacher des politiques).
6. Pour réduire le nombre de politiques disponibles à attacher, pour Filtre, tapez `SSM`.
7. Dans la liste des politiques, cochez la case située à côté d'AmazonSSM ManagedInstanceCore. Utilisez cette stratégie pour fournir les autorisations minimales nécessaires pour utiliser Systems Manager.

   Pour de plus amples informations sur d'autres stratégies que vous pouvez fournir pour les opérations Systems Manager, veuillez consulter Créer un rôle de profil d'instance IAM pour Systems Manager dans le Guide de l'utilisateur AWS Systems Manager .
8. Choisissez Attach Policy (Attacher une politique).

Utilisation de la fonctionnalité Exécuter la commande

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 .

Exemple : Pour mettre à jour le logiciel de l'instance de conteneur avec la fonctionnalité Exécuter la commande

Parmi les cas d'utilisation courants pour la fonctionnalité Exécuter la commande, citons la mise à jour simultanée du logiciel de l'instance sur l'ensemble de votre parc d'instances de conteneur.

1. Attacher des stratégies gérées Systems Manager à votre `ecsInstanceRole`.
2. Vérifier que SSM Agent est installé sur vos instances de conteneur. Pour plus d'informations, veuillez consulter Installer manuellement SSM Agent sur les instances EC2 pour Linux.
3. Ouvrez la console Systems Manager à l'adresse https://console.aws.amazon.com/systems-manager.
4. Dans le panneau de navigation gauche, choisissez Run Command (Exécuter la commande), puis Run Ccmmand (Exécuter une commande).
5. Pour Command document (Document de commande), sélectionnez `AWS-RunShellScript`.
6. Dans la section Commandes, entrez la ou les commandes à envoyer à vos instances de conteneur. Dans cet exemple, la commande suivante met à jour le logiciel de l'instance :

   ```
   $ yum update -y
   ```

7. Dans la section Target instances (Instances cibles), sélectionnez les cases en regard des instances de conteneur où vous souhaitez exécuter la commande de mise à jour.
8. Choisissez Run (Exécuter) pour envoyer la commande aux instances spécifiées.
9. (Facultatif) Choisissez l'icône d'actualisation pour surveiller le statut de la commande.
10. (Facultatif) Dans Targets and ouput (Cibles et sortie), choisissez le bouton en regard de l'ID de l'instance, puis choisissez View output (Afficher la sortie).

Configuration du proxy HTTP pour les instances de conteneur 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 Linux Amazon ECS afin qu'elle utilise un proxy HTTP, définissez les variables suivantes dans les fichiers appropriés lors du lancement (avec les données utilisateur Amazon EC2). Vous pouvez également modifier manuellement le fichier de configuration et redémarrer l'agent par la suite.

`/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`

Utilisez la valeur `169.254.169.254,169.254.170.2,/var/run/docker.sock` pour filtrer des métadonnées d'instance EC2, les rôles IAM et le trafic du démon Docker en provenance 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"`

Utilisez la valeur `169.254.169.254,169.254.170.2,/var/run/docker.sock` pour filtrer des métadonnées d'instance EC2, les rôles IAM et le trafic du démon Docker en provenance 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`

Utilisez la valeur `169.254.169.254,169.254.170.2,/var/run/docker.sock` pour filtrer des métadonnées d'instance EC2, les rôles IAM et le trafic du démon Docker en provenance 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"`

Utilisez la valeur `169.254.169.254` pour filtrer les métadonnées d'instance EC2 issues du 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`

Utilisez la valeur `169.254.169.254` pour filtrer les métadonnées d'instance EC2 issues du 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.

Example Script de données utilisateur proxy HTTP Amazon Linux

L'exemple de script de données utilisateur `cloud-boothook` ci-dessous configure l'agent de conteneur Amazon ECS, `ecs-init`, le démon Docker et yum de manière à 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 Lancement d'une instance de conteneur Amazon ECS Linux et dans Step 6.g. Copiez-collez ensuite le script `cloud-boothook` ci-dessous dans le champ User data (Données utilisateur) (veillez à remplacer les valeurs de l'exemple en rouge par vos propres informations de proxy et de cluster).

> **Note**
>
> Le script de données utilisateur ci-dessous prend uniquement en charge les variantes Amazon Linux 2 et Amazon Linux AMI de l'AMI optimisée pour Amazon ECS.

```
#cloud-boothook
# Configure Yum, the Docker daemon, and the ECS agent to use an HTTP proxy

# Specify proxy host, port number, and ECS cluster name to use
PROXY_HOST=10.0.0.131
PROXY_PORT=3128
CLUSTER_NAME=proxy-test

if grep -q 'Amazon Linux release 2' /etc/system-release ; then
    OS=AL2
    echo "Setting OS to Amazon Linux 2"
elif grep -q 'Amazon Linux AMI' /etc/system-release ; then
    OS=ALAMI
    echo "Setting OS to Amazon Linux AMI"
else
    echo "This user data script only supports Amazon Linux 2 and Amazon Linux AMI."
fi

# Set Yum HTTP proxy
if [ ! -f /var/lib/cloud/instance/sem/config_yum_http_proxy ]; then
    echo "proxy=http://$PROXY_HOST:$PROXY_PORT" >> /etc/yum.conf
    echo "$$: $(date +%s.%N | cut -b1-13)" > /var/lib/cloud/instance/sem/
config_yum_http_proxy
fi

# Set Docker HTTP proxy (different methods for Amazon Linux 2 and Amazon Linux AMI)
# Amazon Linux 2
if [ $OS == "AL2" ] && [ ! -f /var/lib/cloud/instance/sem/config_docker_http_proxy ];
 then
    mkdir /etc/systemd/system/docker.service.d
    cat <<EOF > /etc/systemd/system/docker.service.d/http-proxy.conf
```

```
[Service]
Environment="HTTP_PROXY=http://$PROXY_HOST:$PROXY_PORT/"
Environment="HTTPS_PROXY=https://$PROXY_HOST:$PROXY_PORT/"
Environment="NO_PROXY=169.254.169.254,169.254.170.2"
EOF
    systemctl daemon-reload
    if [ "$(systemctl is-active docker)" == "active" ]
    then
        systemctl restart docker
    fi
    echo "$$: $(date +%s.%N | cut -b1-13)" > /var/lib/cloud/instance/sem/
config_docker_http_proxy
fi
# Amazon Linux AMI
if [ $OS == "ALAMI" ] && [ ! -f /var/lib/cloud/instance/sem/config_docker_http_proxy ];
 then
    echo "export HTTP_PROXY=http://$PROXY_HOST:$PROXY_PORT/" >> /etc/sysconfig/docker
    echo "export HTTPS_PROXY=https://$PROXY_HOST:$PROXY_PORT/" >> /etc/sysconfig/docker
    echo "export NO_PROXY=169.254.169.254,169.254.170.2" >> /etc/sysconfig/docker
    echo "$$: $(date +%s.%N | cut -b1-13)" > /var/lib/cloud/instance/sem/
config_docker_http_proxy
fi

# Set ECS agent HTTP proxy
if [ ! -f /var/lib/cloud/instance/sem/config_ecs-agent_http_proxy ]; then
    cat <<EOF > /etc/ecs/ecs.config
ECS_CLUSTER=$CLUSTER_NAME
HTTP_PROXY=$PROXY_HOST:$PROXY_PORT
NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock
EOF
    echo "$$: $(date +%s.%N | cut -b1-13)" > /var/lib/cloud/instance/sem/config_ecs-
agent_http_proxy
fi

# Set ecs-init HTTP proxy (different methods for Amazon Linux 2 and Amazon Linux AMI)
# Amazon Linux 2
if [ $OS == "AL2" ] && [ ! -f /var/lib/cloud/instance/sem/config_ecs-init_http_proxy ];
 then
    mkdir /etc/systemd/system/ecs.service.d
    cat <<EOF > /etc/systemd/system/ecs.service.d/http-proxy.conf
[Service]
Environment="HTTP_PROXY=$PROXY_HOST:$PROXY_PORT/"
Environment="NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock"
EOF
```

```
    systemctl daemon-reload
    if [ "$(systemctl is-active ecs)" == "active" ]; then
        systemctl restart ecs
    fi
    echo "$$: $(date +%s.%N | cut -b1-13)" > /var/lib/cloud/instance/sem/config_ecs-
init_http_proxy
fi
# Amazon Linux AMI
if [ $OS == "ALAMI" ] && [ ! -f /var/lib/cloud/instance/sem/config_ecs-
init_http_proxy ]; then
    cat <<EOF > /etc/init/ecs.override
env HTTP_PROXY=$PROXY_HOST:$PROXY_PORT
env NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock
EOF
    echo "$$: $(date +%s.%N | cut -b1-13)" > /var/lib/cloud/instance/sem/config_ecs-
init_http_proxy
fi
```

Mise à jour de l'agent de conteneur Amazon ECS

L'agent de conteneur Amazon ECS doit parfois être mis à jour afin d'obtenir les correctifs de bogues et les nouvelles fonctions. 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.

Rubriques

- Vérification de la version d'agent de conteneur Amazon ECS
- Mise à jour de l'agent de conteneur Amazon ECS sur une AMI optimisée pour Amazon ECS
- Mise à jour manuelle de l'agent de conteneur Amazon ECS (pour des AMI non optimisées pour Amazon ECS)

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 page Cluster : ***name*** (Cluster : nom), 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 des AMI non optimisées pour Amazon ECS).

   > **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

plus d'informations, consultez 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 plus d'informations, consultez 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 | python -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 plus d'informations, consultez AMI optimisée pour Amazon ECS.

- 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 plus d'informations, consultez Pour mettre à jour le package `ecs-init` sur une AMI optimisée pour Amazon ECS.
- Mettez à jour l'agent de conteneur avec le fonctionnement de l'`UpdateContainerAgent`API, soit via la console, soit avec les AWS CLI kits de AWS développement logiciel (SDK). Pour plus d'informations, consultez 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
> ```

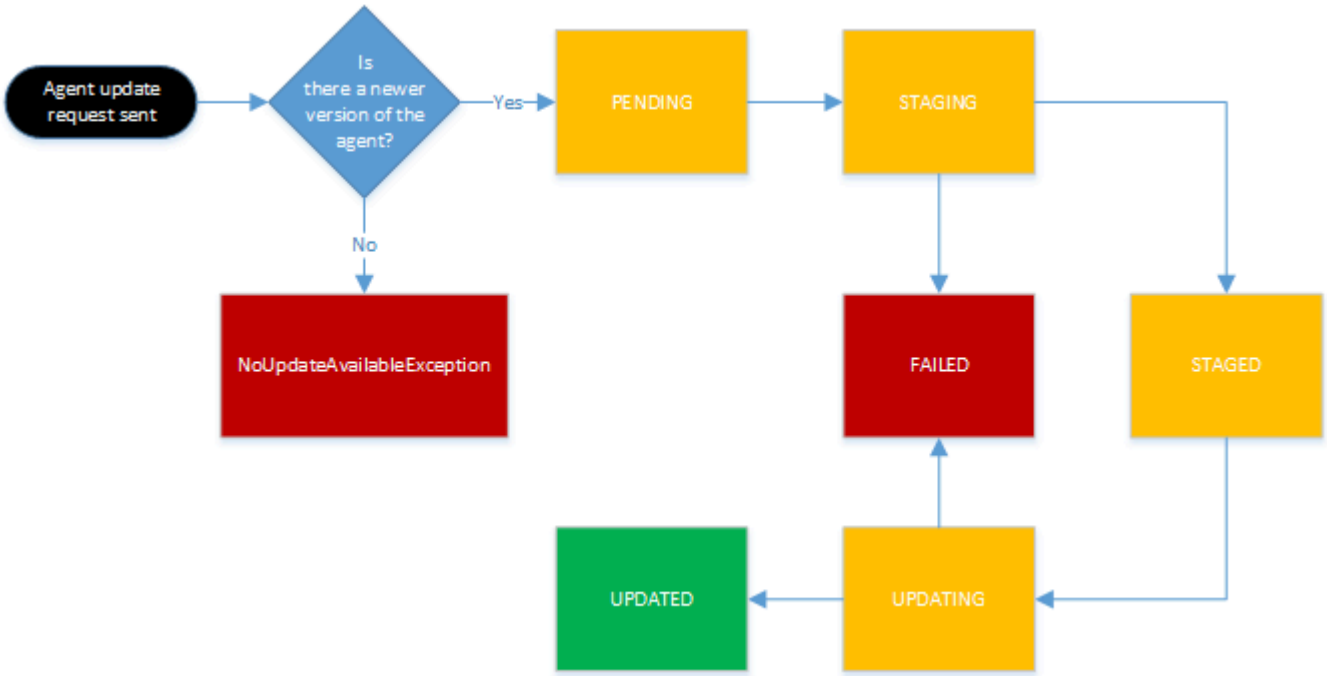
- AMI Amazon Linux optimisée pour Amazon ECS :

```
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 des AMI non optimisées pour Amazon ECS). 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 par le biais de la console, soit à l' AWS CLI aide des kits de AWS développement logiciel (SDK). Amazon ECS vérifie votre version d'agent actuelle par rapport à la dernière version d'agent disponible, et si une mise à jour est possible, le processus de mise à jour évolue comme illustré dans le diagramme ci-dessous. 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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 page Cluster : ***name*** (Cluster : nom), 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 des AMI non optimisées pour Amazon ECS)

Pour mettre à jour manuellement l'agent de conteneur Amazon ECS (pour des AMI non optimisées pour Amazon ECS)

> **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 vos variables d'environnement d'agent dans Amazon S3. Elles pourront être téléchargées dans vos instances de conteneur lors du lancement à l'aide des données utilisateur Amazon EC2. 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 d'instance de conteneur dans Simple Storage Service (Amazon S3) et Authentification de registre privé pour les tâches.

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. Pour plus d'informations, consultez Automatically start containers et Restart policies dans la documentation Docker. 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 avec SELinux activé exigent l'option `--privileged` dans votre commande docker run. En outre, pour les instances de conteneurs avec SELinux activé, nous vous recommandons d'ajouter l'option `:Z` aux montages de volume `/log` et `/data`. 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`. Prenez la mesure suivante si vous rencontrez des difficultés pour exécuter l'agent Amazon ECS sur une instance de conteneur activée pour SELinux :
>
> - 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.

Gestion des instances de conteneur Windows

La gestion des instances de conteneur Windows comprend :

- Lancement d'une instance de conteneur
- Amorçage d'une instance de conteneur
- Connexion à votre instance de conteneur
- Utilisation d'un proxy HTTP pour l'agent de conteneur et le démon Docker
- Annulation de l'enregistrement d'une instance de conteneur

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.

Lancement d'une instance de conteneur Amazon ECS Windows

Vos instances de conteneur Amazon ECS sont créées à l'aide de la console Amazon EC2. 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 Lancement d'une instance à l'aide du nouvel assistant de lancement d'instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.

Nouvel assistant de lancement d'instance Amazon EC2

Vous pouvez utiliser le nouvel assistant Amazon EC2 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.

Paramètres pour la configuration d'instance

- Commencer le lancement de l’instance
- 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

Commencer le lancement de l’instance

1. Ouvrez la console Amazon EC2 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 ressources Amazon EC2 peuvent être partagées entre des régions, contrairement à d’autres ressources.
3. Sur le tableau de bord de la console Amazon EC2, sélectionnez Launch instance (Lancer une 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 les dernières AMI optimisées pour Amazon ECS et leurs valeurs, veuillez consulter AMI optimisées pour Amazon ECS Windows.

Utilisez la barre de recherche pour trouver une AMI optimisée pour Amazon ECS adaptée publiée par. AWS

1. Selon vos besoins, saisissez l'une des AMI suivantes dans la barre de recherche et appuyez sur la touche Enter (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 Sélection d'une Amazon Machine Image (AMI) sélectionnez l'onglet AMI de 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 Configuration du proxy HTTP pour les instances de conteneur 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 plus d'informations, consultez 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 utilisateur Amazon EC2 sont exécutés une seule fois, au 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 User data (Données utilisateur), en remplaçant *`your_cluster_name`* 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>
```

Ancien assistant de lancement d'instance Amazon EC2

Pour lancer une instance de conteneur

1. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Sur le tableau de bord EC2, sélectionnez Launch instance (Lancer une instance).
4. Sur la page Choose an Amazon Machine Image (AMI) (Choisir une Amazon Machine Image (AMI)), effectuez les étapes suivantes :

   a. Choisissez AWS Marketplace.

   b. Choisissez une AMI pour votre instance de conteneur. Vous pouvez rechercher l'une des AMI optimisées pour Amazon ECS, par exemple Windows_2019_Full_ECS_Optimized. Si vous ne choisissez pas une AMI optimisée pour Amazon ECS, vous devez suivre les procédures décrites dans Installation de l'agent de conteneur Amazon ECS.
5. Sur la page Choose an Instance Type (Choisir un type d'instance), vous pouvez sélectionner la configuration matérielle de votre instance. Le type d'instance `t2.micro` est sélectionné par

défaut. Le type d'instance que vous sélectionnez détermine les ressources disponibles pour les tâches à exécuter.

Lorsque vous avez terminé, sélectionnez Next: Configure Instance Details (Suivant : Configurer les détails d'instance).

6. Dans la page Configure Instance Details (Configurer les détails d'instance), complétez les étapes suivantes :

   a. Définissez le champ Number of instances (Nombre d'instances) en fonction du nombre d'instances de conteneur à ajouter à votre cluster.

   b. (Facultatif) Pour utiliser des instances Spot, pour Purchasing option (Option d'achat), cochez la case en regard de Request Spot Instances (Demander des instances 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.

   c. Pour Network (Réseau), choisissez le VPC dans lequel lancer votre instance de conteneur.

   d. Pour Subnet (Sous-réseau), choisissez un sous-réseau à utiliser ou conservez l'option par défaut pour choisir le sous-réseau par défaut dans une zone de disponibilité.

   e. Définissez le champ Auto-assign Public IP (Attribuer automatiquement l'adresse IP publique) pour indiquer si vous souhaitez que votre instance soit accessible ou non à partir du réseau Internet public. 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 service. 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, veuillez consulter 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 Configuration du proxy HTTP pour les instances de conteneur Linux dans ce guide. Pour plus d'informations, consultez the section called "Créer un Virtual Private Cloud".

f. Sélectionnez le rôle IAM de votre 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 plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.

g. 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 utilisateur Amazon EC2 sont exécutés une seule fois, au 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 User data (Données utilisateur), en remplaçant *`your_cluster_name`* 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>
```

h. Choisissez Next: Add Storage (Suivant : Ajouter le stockage).

7. Sur la page Add Storage (Ajouter le stockage), configurez le stockage pour votre instance de conteneur.

   Vous pouvez augmenter ou diminuer la taille des volumes pour votre instance afin de répondre aux besoins de votre application.

   Lorsque vous avez terminé de configurer vos volumes, choisissez Next: Add Tags (Suivant : Ajouter des étiquettes).

8. Sur la page Add Tags (Ajouter des étiquettes), spécifiez les étiquettes en fournissant les combinaisons clé et valeur pour l'instance de conteneur. Choisissez Add another tag (Ajouter une autre étiquette) pour ajouter plusieurs étiquettes à votre instance de conteneur. Pour plus d'informations sur les étiquettes de ressource, consultez Balisage des ressources Amazon ECS.

   Choisissez Next: Configure Security Group (Suivant : Configurer le groupe de sécurité) une fois que vous avez terminé.

9. Sur la page Configure Security Group (Configurer le groupe 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. Sélectionnez ou créez un groupe de sécurité de la façon suivante, puis choisissez Review and Launch (Vérifier et lancer).

10. Sur la page Review Instance Launch (Examiner le lancement de l'instance), sous Security Groups (Groupes de sécurité), vous verrez que l'assistant a créé et sélectionné un groupe de

sécurité pour vous. Au lieu de cela, sélectionnez le groupe de sécurité que vous avez créé à l'étape Configurer l'utilisation d'Amazon ECS en procédant comme suit :

a. Sélectionnez Edit security groups (Modifier les groupes de sécurité).

b. Sur la page Configure Security Group (Configurer le groupe de sécurité), choisissez l'option Select an existing security group (Sélectionnez un groupe de sécurité existant).

c. Sélectionnez le groupe de sécurité que vous avez créé pour votre instance de conteneur dans la liste des groupes de sécurité existants, puis choisissez Review and Launch (Vérifier et lancer).

11. Sur la page Review Instance Launch (Examiner le lancement d'instance), sélectionnez Launch (Lancer).
12. Dans la boîte de dialogue Sélectionnez une paire de clés existante ou créez une nouvelle paire de clés, sélectionnez Choisir une paire de clés existante, puis sélectionnez la paire de clés créée lors de la configuration.

    Lorsque vous êtes prêt, cochez la case de confirmation, puis sélectionnez Launch Instances (Lancer des instances).
13. 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.
14. Sur l'écran Instances, vous pouvez afficher le statut de votre instance. Il suffit de peu de temps pour lancer une instance. Lorsque vous lancez une instance, son état initial est `pending`. Une fois que l'instance a démarré, son état devient `running` et elle reçoit un nom DNS public. Si la colonne DNS public est masquée, choisissez l'icône Afficher/Masquer, DNS public.

Utilisation des instances Spot

Une instance Spot est une instance Amazon EC2 non utilisée qui est disponible à un prix inférieur au prix à la demande. Comme les Instances Spot vous permettent de demander des instances EC2 inutilisées avec de fortes remises, vous pouvez réduire considérablement vos coûts Amazon EC2. Le prix horaire d'une instance Spot est appelé prix spot. Le prix Spot de chaque type d'instance dans chaque zone de disponibilité est défini par Amazon EC2 et varie en fonction de l'offre et de la demande à long terme pour les Instances Spot. Pour plus d'informations, veuillez consulter Instances Spot dans le Guide de l'utilisateur Amazon EC2 pour instances Windows.

Vous pouvez enregistrer des instances Spot auprès de vos clusters Amazon ECS. Pour plus d'informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

Drainage des instances Spot

Amazon EC2 résilie, arrête ou met en veille prolongée votre instance Spot lorsque le prix Spot dépasse le prix maximum de votre demande ou lorsque la capacité n'est plus disponible. Amazon EC2 communique un avis d'interruption d'instance Spot, qui donne à l'instance un avertissement deux minutes avant qu'elle soit interrompue. 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 devez définir le paramètre `ECS_ENABLE_SPOT_INSTANCE_DRAINING` avant de démarrer l'agent de conteneur. Utilisez les commandes suivantes pour activer manuellement le drainage des instances Spot. 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 plus d’informations, consultez the section called “Lancement d'une instance de conteneur”.

Action d'amorçage des instances de conteneur Windows avec les données utilisateur d'Amazon EC2

Lorsque vous lancez une instance de conteneur Amazon ECS, vous avez la possibilité de transmettre les données utilisateur à cette instance. Les données peuvent être utilisées pour effectuer des tâches de configuration automatisées courantes et exécuter des scripts au 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 de cloud, des scripts shell et des directives `cloud-init`. 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 lors de l'utilisation de l'Amazon EC2 Launch Wizard. Pour plus d’informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

Rubriques

- Données utilisateur Windows par défaut
- Données utilisateur d'installation d'un agent Windows
- Rôles IAM Windows pour les tâches

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 référentiel Amazon ECS Container Agent sur GitHub.

> **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 plus d'informations, consultez Création d'un cluster pour le type de lancement Fargate et External à l'aide de la console.

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>
```

Rôles IAM Windows pour les tâches

Consultez les exemples Windows suivants concernant l'action d'amorçage des rôles de tâche IAM.

- Configuration supplémentaire des rôles IAM pour les tâches Windows
- Rôles IAM pour le script d'amorçage du conteneur de tâches

Connectez-vous à votre instance Windows de conteneur

Vous pouvez vous connecter à vos instances Windows pour effectuer des tâches administratives de base, telles que l'installation ou la mise à jour de logiciels ou l'accès aux journaux de diagnostics. Pour vous connecter à votre instance à l'aide du protocole RDP (Remote Desktop Protocol), veuillez remplir les conditions suivantes.

- Les instances Amazon EC2 créées à partir de la plupart des AMI Windows vous permettent de vous connecter à l'aide du protocole RDP (Remote Desktop Protocol). Le protocole RDP (Remote Desktop Protocol) vous permettent de vous connecter à l'instance et de l'utiliser comme s'il s'agissait d'un ordinateur physique. Il est disponible sur la plupart des éditions de Windows et disponible pour Mac OS.
- Votre instance Windows doit avoir été lancée avec une paire de clés Amazon EC2 valide. Les instances Amazon EC2 n'ont pas de mot de passe, utilisez une paire de clés pour l'accès via RDP. Si vous n'avez pas spécifié de paire de clés lorsque vous avez lancé votre instance, vous ne pouvez pas vous connecter à l'instance. Pour plus d'informations, consultez the section called “Lancement d'une instance de conteneur”.
- Vérifiez que le groupe de sécurité associé à votre instance autorise le trafic RDP entrant (port 3389) à partir de votre adresse IP. Le groupe de sécurité par défaut n'autorise pas le trafic

RDP entrant par défaut. Pour plus d'informations, consultez Autoriser le trafic entrant pour vos instances Windows dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.

1. Trouvez l'adresse IP publique ou DNS pour votre instance de conteneur.
   a. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
   b. Dans le panneau de navigation, choisissez Clusters, puis sélectionnez le cluster qui héberge l'instance.
   c. Sur la page Cluster : ***name*** (Cluster : nom), choisissez l'onglet Infrastructure.
   d. Sous Constainer instances (Instances de conteneur), sélectionnez l'ID d'instance
   e. Sur la page Instances (Instances), enregistrez l'adresse Public IP (IP publique) ou Public DNS (DNS public) de votre instance.
2. Trouvez le nom d'utilisateur par défaut de votre AMI d'instance de conteneur.
3. Vous pouvez vous connecter à votre instance à l'aide de RDP. Pour plus d'informations, consultez Se connecter à votre instance Windows à l'aide de RDP dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.

Configuration du proxy HTTP pour les instances de conteneur Windows

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 Windows afin qu'elle utilise un proxy HTTP, définissez les variables suivantes lors du lancement (avec les données utilisateur Amazon EC2).

```
[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 `NO_PROXY` sur `169.254.169.254,169.254.170.2,\\.\pipe\docker_engine` pour filtrer les métadonnées d'instance EC2, les rôles IAM pour les tâches et le trafic du démon Docker en provenance 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 plus d'informations, consultez Configuration supplémentaire des rôles IAM pour les tâches 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>
```

Annuler l'enregistrement d'une instance de conteneur sauvegardée par Amazon EC2

> **Important**
>
> Cette rubrique concerne uniquement les instances de conteneur créées dans Amazon EC2. Pour plus d'informations sur l'annulation de l'enregistrement des instances externes, veuillez consulterAnnuler l'enregistrement d'une instance externe.

Lorsque vous avez terminé avec une instance de conteneur sauvegardée par Amazon EC2, vous pouvez en annuler l'enregistrement dans 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 de plus amples informations, veuillez consulter Résilier une instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux et Lancement 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.
- L'annulation de l'enregistrement d'une instance de conteneur supprime l'instance d'un cluster, mais ne résilie pas 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 Résilier votre instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 page Cluster : ***name*** (Cluster : nom), 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 page Container Instance : ***id*** (Instance de conteneur : id), choisissez Deregister (Annuler l'enregistrement).
7. Dans le message de confirmation, Annuler l’inscription.
8. Si vous n'avez plus besoin de l'instance de conteneur, résiliez l'instance Amazon EC2 sous-jacente. Pour plus d'informations, consultez Résilier votre instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

Drainage d'instance géré par Amazon ECS

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 instances Amazon EC2, 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 et est activé lors de la création ou de la mise à jour des 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 instances Amazon EC2 dans votre groupe Auto Scaling au lieu de le faire manuellement par lots. Cela est particulièrement utile lorsque vous devez remplacer un grand nombre d'instances. L'actualisation d'une instance est initiée via la console Amazon EC2 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 le groupe Auto Scaling avec les fournisseurs de capacité Amazon ECS, qui facilitent le dimensionnement automatique des clusters, 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 stack updates - Vous pouvez ajouter un `UpdatePolicy` attribut à votre pour effectuer des mises AWS CloudFormation 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 de capacité Amazon EC2. 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 qui se termine, tout comme il draine une instance autre que 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.

Crochets liés au cycle de vie d'Amazon EC2 Auto Scaling avec 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 Lifecycle possèdent 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 s'exécuteront. 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.

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 restera inchangée. Vous devrez 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 jusqu'à ce 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

Comme le drainage des instances géré par Amazon ECS facilite le drainage progressif des instances Amazon EC2, il empêche vos applications d'être perturbées par tout événement de résiliation. Le drainage géré des instances facilite l'arrêt progressif des tâches de service Amazon ECS pour les instances en cours d'extension, même si la résiliation gérée est désactivée.

Le tableau suivant récapitule le comportement des différentes combinaisons de terminaison gérée et de drainage géré.

Résiliation gérée	Drainage géré	Outcome
Activées	Activées	Amazon ECS empêche les instances Amazon EC2 qui exécutent des tâches d'être interrompues par des événements de scale-in. Toutes les instances en cours de résiliation, telles que celles pour lesquelles la protection contre

Résiliation gérée	Drainage géré	Outcome
		la résiliation n'est pas définie, celles qui ont subi une interruption ponctuelle ou qui sont forcées par une actualisation de l'instance, sont correctement drainées.

Résiliation gérée	Drainage géré	Outcome
Désactivées	Activées	Amazon ECS ne protège pas les instances Amazon EC2 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 empêche les instances Amazon EC2 qui exécutent des tâches d'être interrompues par 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	Les instances Amazon EC2 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'instance Amazon ECS 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 instances Amazon EC2 en cours de résiliation pour de nombreuses raisons, et pas simplement pour une interruption ponctuelle. Par exemple, vous pouvez utiliser le rééquilibrage des capacités d'Amazon EC2 Auto Scaling pour remplacer de manière proactive les instances Spot présentant un risque élevé

d'interruption, et le drainage géré des instances permet d'arrêter progressivement les instances Spot remplacées. Lorsque vous utilisez le drainage d'instance géré, il n'est pas nécessaire d'activer le drainage d'instance Spot séparément. `ECS_ENABLE_SPOT_INSTANCE_DRAINING` Dans ASG, les données utilisateur sont donc redondantes.

Résolution des problèmes liés au drainage des instances gérées par Amazon ECS

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 qui peuvent être récupérés sur la console et l'API Amazon ECS en cas de défaillance lors du provisionnement des ressources de drainage des instances gérées.
- EventBridge comporte des mécanismes de réessai intégrés pour atténuer les défaillances temporaires.

Utilisation du drainage d'instance géré par Amazon ECS

Vous activez 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 d'instance géré 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.

```
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": []
        }
    ]
}
```

Planification de 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.

Planificateur de service

Le planificateur de services convient aux services et applications apatrides 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 peut replanifier des tâches.

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. Si l'état de santé de la tâche de remplacement est `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).

> **Note**
>
> Ce comportement ne s'applique pas aux tâches exécutées et gérées par les services qui utilisent la mise à jour propagée comme type de déploiement. Lors d'une mise à jour propagée, le planificateur de service arrête d'abord les tâches défectueuses, puis lance les tâches de remplacement.

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 plus d'informations, consultez Replica.
- `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 plus d'informations, consultez Démon.

  > **Note**
  >
  > Les tâches Fargate ne prennent pas en charge la stratégie de planification `DAEMON`.

Le planificateur de service peut aussi s'assurer que les tâches sont enregistrées 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 déploiement d'une nouvelle définition de tâche ou la modification du nombre de tâches souhaitées en cours d'exécution. Par défaut, le planificateur de service répartit les tâches entre plusieurs zones 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. Pour plus d'informations, consultez Services Amazon ECS.

Tâches autonomes

Vous pouvez exécuter une tâche autonome lorsque des processus tels que des tâches par lots 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. Vous pouvez également utiliser `RunTask` afin de personnaliser la façon dont le planificateur place les tâches à l'aide des stratégies et contraintes de placement des tâches.

Planificateurs personnalisés

Avec Amazon ECS, vous pouvez créer vos propres planificateurs ou utiliser des planificateurs tiers. Pour plus d'informations, consultez Comment créer un planificateur personnalisé pour Amazon ECS. Les planificateurs personnalisés utilisent l'opération StartTaskAPI pour placer des tâches sur des instances de conteneur spécifiques au sein de votre cluster.

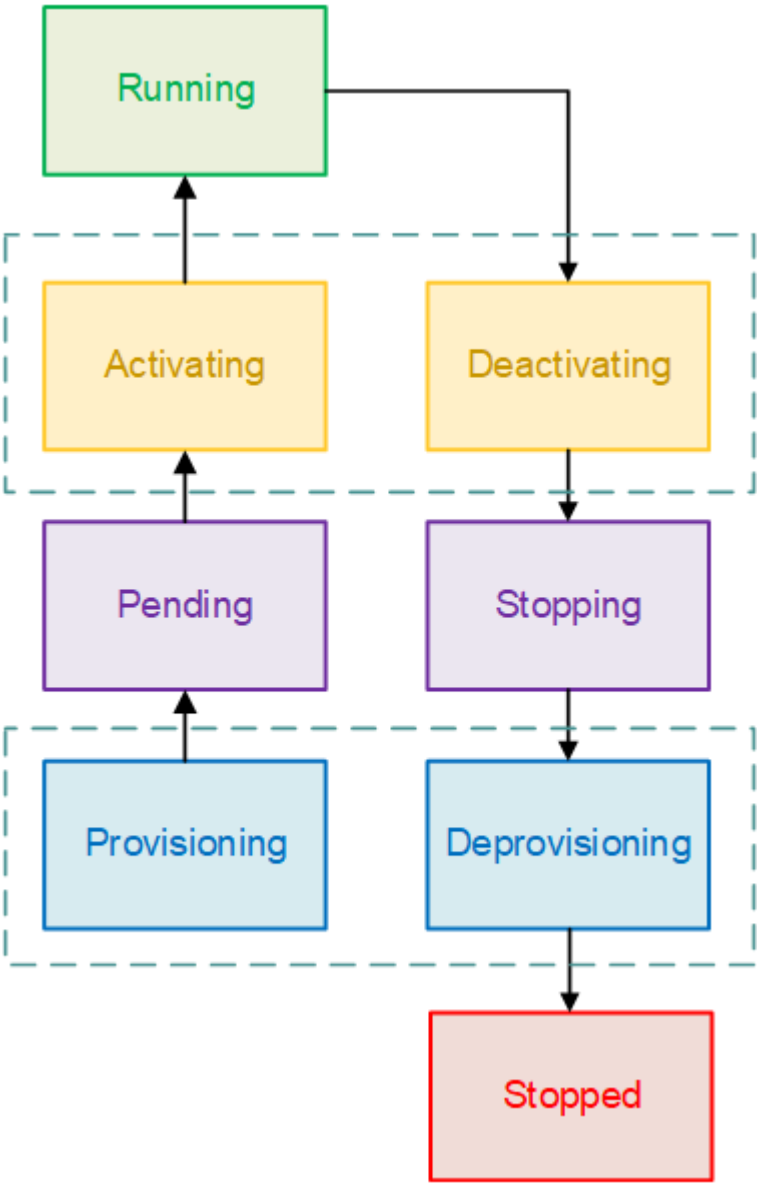
> **Note**
>
> Les planificateurs personnalisés sont uniquement compatibles avec les tâches hébergées sur les instances EC2. Si vous utilisez Amazon ECS sur Fargate, StartTask l'API ne fonctionne pas.

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`. Par exemple, pour les tâches qui disposent d'une configuration de découverte de service, les ressources de découverte de service doivent être créées. Pour les tâches qui font partie d'un service qui est configuré pour utiliser plusieurs groupes cibles Elastic Load Balancing, l'enregistrement du groupe cible se produit au cours de cet état.

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. Pour les tâches qui font partie d'un service qui est configuré pour utiliser plusieurs groupes cibles Elastic Load Balancing, l'annulation de l'enregistrement du groupe cible se produit au cours de 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 enverra le signal `SIGTERM` pour informer que l'application doit se terminer et s'arrêter, puis il enverra un `SIGKILL` après avoir attendu la durée `StopTimeout` définie dans la définition de 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.

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.

Type de lancement EC2

Pour les tâches qui utilisent le type de lancement EC2, 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 dans les services. Pour plus d'informations, consultez Planification de 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 lancement EC2, 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 dans les services. Pour plus d'informations, consultez Planification de 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 la résiliation de la tâche voit cette tâche résiliée.	• cpu • memory	
`random`	Non utilisé	

type	Valeurs de champ valides	
Les tâches sont placées au hasard.		
`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` (ouhost, 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 plus d'informations, consultez 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 plus d'informations, consultez 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.

```
"placementConstraint": [
```

```
    {
        "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` Place chaque tâche sur une instance de conteneur différente. **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	• Exécution d'une tâche RunTask • Création d'un nouveau service CreateService,	

Constraint type (Type de contrainte)	Peut être spécifié quand	
plus d'informations, consultez la section Présentation de la sécurité de AWS Fargate.		
`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 de`placementConstraint`.

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 instance EC2 exécutée sur la même version en utilisant la contrainte de placement : `memberOf(attribute:ecs.os-family == WINDOWS_SERVER_<OS_Release>_<FULL or CORE>)`. Pour plus d'informations, consultez the section called “Extraction des métadonnées d'AMI optimisée 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 plus d'informations, consultez Jonction d'interface réseau Elastic.

`ecs.outpost-arn`

Si cet attribut existe, il contient l'Amazon Resource Name (ARN) de l'Outpost. Pour plus d'informations, consultez 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 plus d'informations, consultez Instances externes (Amazon ECS Anywhere).

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 plus d'informations, consultez 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 Exécuter une application en tant que tâche Amazon ECS et Création d'un service à l'aide de la console. Vous pouvez également appliquer un filtre de groupe lorsque vous répertoriez les instances de conteneur. Pour plus d'informations, consultez Filtrage par attribut à l'aide de la console.

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 plus d'informations, consultez Attributs de l'instance de conteneur Amazon ECS.

`ec2InstanceId`

Sélectionnez les instances de conteneur par leur ID d'instance Amazon EC2.

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 plus d'informations, consultez Tâches Amazon ECS liées au groupe .

Opérateur

Opérateur de comparaison. Les opérateurs suivants sont pris en charge.

Opérateur	Description
==, equals	Égalité de chaînes
!=, 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'instance Amazon EC2

L'expression suivante sélectionne des instances avec les ID d'instance Amazon EC2 suivants.

```
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 démon sont placées sur l'instance EC2 avant les tâches de service de réplica.

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

```
"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, les API ou les SDK.

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 Amazon EC2. 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 lancement EC2, vous pouvez soit gérer la capacité de votre cluster en enregistrant des instances Amazon EC2 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, veuillez consulter Comment Amazon ECS place les tâches sur les instances de conteneur.

Après avoir trouvé la capacité de placer votre tâche, Amazon ECS fournit les pièces jointes nécessaires (par exemple, des interfaces réseau élastiques (ENI) 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.

Rubriques

- Bonnes pratiques pour améliorer le temps de lancement des tâches Amazon ECS
- Exécuter une application en tant que tâche Amazon ECS
- Exécutez des tâches Amazon ECS selon des plannings à l'aide d'Amazon EventBridge Scheduler
- Arrêter une tâche Amazon ECS

Bonnes pratiques pour améliorer 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 lancement EC2, 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 aucune 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 ont généralement des tailles d'image de conteneur importantes (des

dizaines de Go). 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 les API Amazon EC2, 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 plus d'informations, consultez the section called "AWSVPC mode".

- 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 `ContainerStartTime` jusqu'au moment où celle-ci est prête à traiter le trafic. 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 plus d'informations, consultez Récupérez les métadonnées Amazon ECS.

- Choisissez un type d'instance optimal (lors de l'utilisation du type de lancement EC2)

  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.

Exécuter 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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 choisissez.

Pour démarrer un service à partir de	Étapes
une page détaillée du cluster...	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).
une 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.

Pour démarrer un service à partir de	Étapes
	c. Dans le menu Déployer, choisissez Exécuter la tâche.

3. (Facultatif) La section Configuration informatique (avancée) vous permet de choisir le mode de distribution de vos tâches. Vous pouvez utiliser une stratégie de fournisseur de capacité ou un type de lancement. Pour utiliser une stratégie de fournisseur de capacité, vous devez configurer vos fournisseurs de capacité au niveau du cluster. Pour plus d'informations, consultez Fournisseurs de capacité Amazon ECS. 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). 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, sélection

Méthode de distribution	Étapes
	nez Use custom (Utiliser une stratégie personnalisée) ou Add capacity provider strategy (Ajouter une stratégie de fournisseur de capacité) et définissez 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 capacité dans une stratégie, le fournisseur de capacité doit être associé au cluster. Pour de plus amples informations sur les stratégies de fournisseurs de capacité, veuillez consulter Fournisseurs de capacité Amazon ECS.

Méthode de distribution	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.

4. Pour Application type (Type d'application), choisissez Task (Tâche).
5. Pour Définition de tâche, choisissez la famille et la révision de définition de 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.

6. Pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer.
7. 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.

> **Important**
>
> Seuls les sous-réseaux privés sont pris en charge pour le mode réseau `awsvpc`. Les tâches ne reçoivent pas les adresses IP publiques. Par conséquent, une passerelle NAT est requise pour l'accès Internet sortant. Par ailleurs, le trafic Internet entrant est acheminé par le biais d'un équilibreur de charge.

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. Pour plus d'informations, consultez Mise en réseau des tâches Fargate dans le Guide de l'utilisateur Amazon Elastic Container Service pour AWS Fargate.

8. 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'opérations d'entrée/sortie (IOPS) que le volume doit fournir. Cette valeur est configurable uniquement 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	

Pour configurer ce type de volume	Faites ceci	
	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. 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. 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	

Pour configurer ce type de volume	Faites ceci	
	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. 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, voirRôle IAM dans l'infrastructure Amazon ECS.	

Pour configurer ce type de volume	Faites ceci	
	i. 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 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.	

Pour configurer ce type de volume	Faites ceci	
	j. 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. 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, soit en fournissant vos propres balises. Si vous souhaitez propager des balises à partir de la définition de tâche, choisissez Définitio	

Pour configurer ce type de volume	Faites ceci	
	n 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'informa tions sur le balisage des volumes Amazon EBS, consultez la section Marquage des volumes Amazon EBS.	

9. (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 plus d'informations, consultez 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.

10. (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 plus d'informations, consultez Rôle IAM de tâche.

       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 d'un rôle et d'une politique IAM pour vos tâches.

    b. Pour Rôle d'exécution de tâche, choisissez un rôle d'exécution de tâche. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

11. (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 de plus amples informations sur la commande Docker run, consultez la Référence Docker Run dans le manuel de référence 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."
  ```

12. (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.
13. Choisissez Créer.

Exécutez des tâches Amazon ECS selon des plannings à l'aide d'Amazon EventBridge Scheduler

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 les ID de sous-réseaux sur lesquels les tâches s'exécutent et les ID de groupe de sécurité pour les sous-réseaux. Pour plus d'informations, veuillez consulter Afficher vos sous-réseaux et Afficher vos 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.

Pour créer une planification à l'aide de la console

1. Ouvrez la console Amazon EventBridge Scheduler à l'adresse https://console.aws.amazon.com/scheduler/home.
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 planifications, choisissez un groupe de planifications dans la liste déroulante. 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 que vous spécifiez à l'aide d'une expression cron ou d'une expression rate.	a. Pour Schedule type (Planifier le type), effectuez l'une des étapes suivantes : • 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 Planifica

Occurrence	Faites ceci...
	tion 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 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 Toutes les API, puis saisissez 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. Pour utiliser un type de lancement, développez Options de calcul, puis sélectionnez Type de lancement. Choisissez ensuite le type de lancement.

      Lorsque le type de lancement Fargate est spécifié, pour Version de plateforme, saisissez la version de plateforme à utiliser. Si vous ne spécifiez aucune plateforme, la version de plateforme `LATEST` est utilisée.

   g. Pour Sous-réseaux, saisissez les ID de sous-réseaux dans lesquels exécuter la tâche.

   h. Pour Groupes de sécurité, saisissez les ID des groupes de sécurité pour le 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 plus d'informations, consultez 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.
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.

Option File d'attente de lettres mortes (DLQ)	Faites ceci...	
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.

Arrêter une tâche Amazon ECS

Si vous n'avez plus besoin de continuer à exécuter une tâche autonome, vous pouvez l'arrêter. La console Amazon ECS permet d'arrêter facilement une ou plusieurs tâches.

Si vous souhaitez arrêter un service, consultezSuppression d'un service avec la console.

Pour arrêter une tâche autonome ()AWS Management Console

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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. b. Sur la page de confirmation de l'arrêt de la tâche, choisissez Arrêter
Déploiement bleu/vert	**Important** Si vous choisissez d'arrêter toutes les tâches à l'aide de la console, Amazon ECS arrête toutes les

Tâches à arrêter	Étapes
	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.

Concepts du planificateur de 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. Si l'état de santé de la tâche de remplacement est `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).

> **Note**
>
> Ce comportement ne s'applique pas aux tâches exécutées et gérées par les services qui utilisent la mise à jour propagée comme type de déploiement. Lors d'une mise à jour propagée, le planificateur de service arrête d'abord les tâches défectueuses, puis lance les tâches de remplacement.

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 Messages d'événement 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 contraintes de placement des tâches afin de personnaliser la façon dont les tâches sont placées. Pour plus d'informations, consultez Replica.
- `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 plus d'informations, consultez Démon.

> **Note**
>
> Les tâches Fargate ne prennent pas en charge la stratégie de planification `DAEMON`.

Démon

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 service évalue également les contraintes de placement des tâches pour les tâches en cours d'exécution et arrête les tâches qui ne respectent pas les contraintes de placement. En utilisant cette stratégie, vous n'avez pas besoin de spécifier un nombre de tâches souhaité, une stratégie de placement des tâches ou d'utiliser des stratégies 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 nouvelle tâche de démon est la première à se lancer sur les instances et la dernière à s'arrêter. 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 garantit que les tâches du démon sont les dernières à s'arrêter après l'arrêt de toutes les tâches de réplica. Le planificateur de service surveille également l'ajout de nouvelles instances de conteneur à votre cluster et leur ajoute les tâches du démon.

Si une configuration de déploiement est spécifiée, le paramètre de pourcentage maximal doit être `100`. La valeur par défaut d'un service démon pour `maximumPercent` est 200 %. La valeur par défaut d'un service démon pour `minimumHealthyPercent` est 0 %.

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. 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.

Replica

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 instances EC2, vous pouvez, si vous le souhaitez, spécifier des stratégies et des contraintes de placement de tâches afin de personnaliser les décisions relatives à cette opération. 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 réplica dépend d'un service démon (par exemple, une tâche de routeur de journaux démon 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 démon sont placées sur l'instance EC2 avant les tâches du service de réplica. Pour plus d'informations, consultez 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.

Concepts du service supplémentaire

- Le cas échéant, vous pouvez exécuter votre service derrière un équilibreur de charge. Pour plus d'informations, consultez Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge.
- Il vous est possible de spécifier une configuration de déploiement pour votre service. Un déploiement est initié par la mise à jour de la définition de tâche d'un service. Pendant un déploiement, le planificateur de service utilise les paramètres de pourcentage minimum d'instances saines et de pourcentage maximal afin de déterminer la stratégie de déploiement. Pour plus d'informations, consultez Paramètres de définition de service.
- Vous pouvez éventuellement configurer votre service afin d'utiliser la découverte de service Amazon ECS service. La découverte de services utilise les AWS Cloud Map API de dénomination automatique pour gérer les entrées DNS pour les tâches de votre service. Cela les rend détectables dans votre VPC. Pour plus d'informations, consultez Découverte de service.
- Lorsque vous supprimez un service, si des tâches sont toujours en cours d'exécution et doivent être annulées, le service passe de l'état `ACTIVE` à `DRAINING`, et le service n'est plus visible dans la console ni dans l'opération d'API `ListServices`. Une fois que toutes les tâches passent à l'état `STOPPING` ou `STOPPED`, l'état du service passe de `DRAINING` à `INACTIVE`. Vous pouvez consulter les services ayant l'état `DRAINING` ou `INACTIVE` en utilisant l'opération d'API

`DescribeServices`. Cependant, à l'avenir, les services `INACTIVE` pourraient être nettoyés et purgés des registres Amazon ECS et les appels de `DescribeServices` sur ces services renverraient alors une erreur `ServiceNotFoundException`.

- La durée de l'intégration correspond à la période après qu'une nouvelle version d'un service ait monté en puissance et que l'ancienne version ait fait l'objet d'une mise à l'échelle horizontale. Pendant cette période, Amazon ECS continue de surveiller l'alarme associée au déploiement. Amazon ECS calcule cette période en fonction de la configuration d'alarme associée au déploiement.

  Le temps de cuisson s'applique uniquement lorsque vous utilisez des CloudWatch alarmes pour détecter les échecs de déploiement. Pour plus d'informations, consultez the section called "Méthodes de détection des échecs".

Meilleures pratiques pour améliorer les délais de déploiement d'Amazon ECS

Lorsque vous utilisez l'option de déploiement des mises à jour continues, vous pouvez modifier les paramètres pour accélérer vos déploiements.

Lorsque vous choisissez le type de déploiement des mises à jour continues, le planificateur de services Amazon ECS remplace toutes les tâches en cours d'exécution par de nouvelles tâches chaque fois qu'un nouveau déploiement de service est lancé. La configuration de déploiement détermine le nombre spécifique de tâches qu'Amazon ECS ajoute ou supprime du service lors d'une mise à jour continue. Voici un aperçu du processus de déploiement :

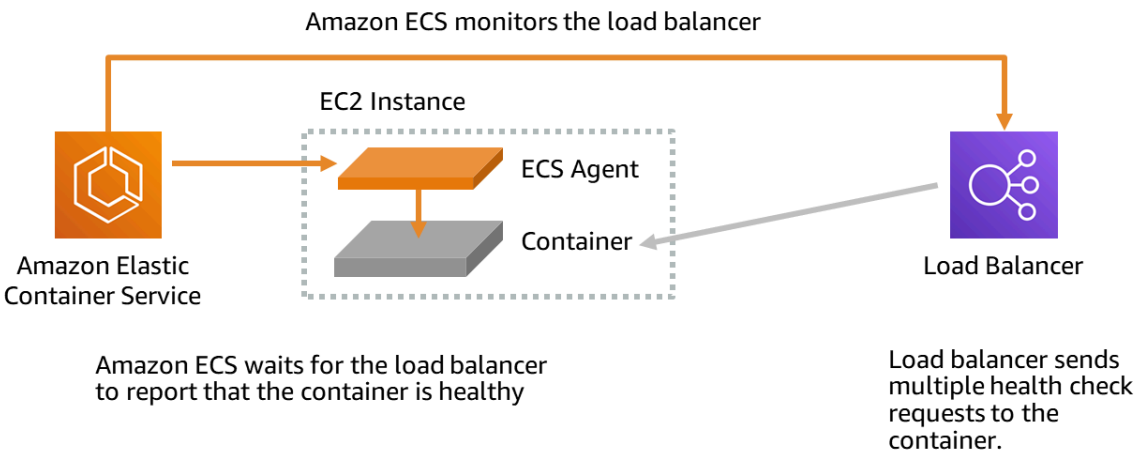
1. Le planificateur démarre votre application.
2. Le planificateur décide ensuite si votre application est prête à recevoir du trafic.
3. Lorsque vous réduisez la taille ou que vous créez une nouvelle version de l'application, le planificateur décide si votre application peut être arrêtée en toute sécurité. Dans le même temps, il doit maintenir la disponibilité de l'application pendant le déploiement des mises à jour continues.

La stratégie de maintien de la disponibilité des tâches peut entraîner des déploiements plus longs que prévu.

Pour accélérer les délais de déploiement, modifiez l'équilibreur de charge par défaut, l'agent Amazon ECS, les options de définition des services et des tâches.

Paramètres de vérification de l'état de l'équilibreur de charge pour accélérer les délais de déploiement d'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 équilibreur de charge vérifie l'état de santé de chaque cible en utilisant les paramètres de contrôle de santé des groupes cibles auprès desquels la cible est enregistrée. Une fois que votre cible est enregistrée, elle doit passer avec succès une seule vérification de l'état pour être considérée comme saine. Le schéma suivant décrit le processus de vérification de l'état de santé de 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 :

- `HealthCheckIntervalSeconds`dans l'API Elastic Load Balancing
- Intervalle sur la console Amazon EC2

- 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 :

  - `HealthyThresholdCount`dans l'API Elastic Load Balancing
  - Seuil sain sur la console Amazon EC2

Avec les paramètres par défaut, le temps total nécessaire pour déterminer l'état de santé 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 le nombre de bilans de santé et l'intervalle entre les contrôles.

- `HealthCheckIntervalSeconds`(nom de l'API Elastic Load Balancing) ou Interval (nom de la console Amazon EC2) : 5
- `HealthyThresholdCount`(nom de l'API Elastic Load Balancing) ou seuil sain (nom de la console Amazon EC2) : 2

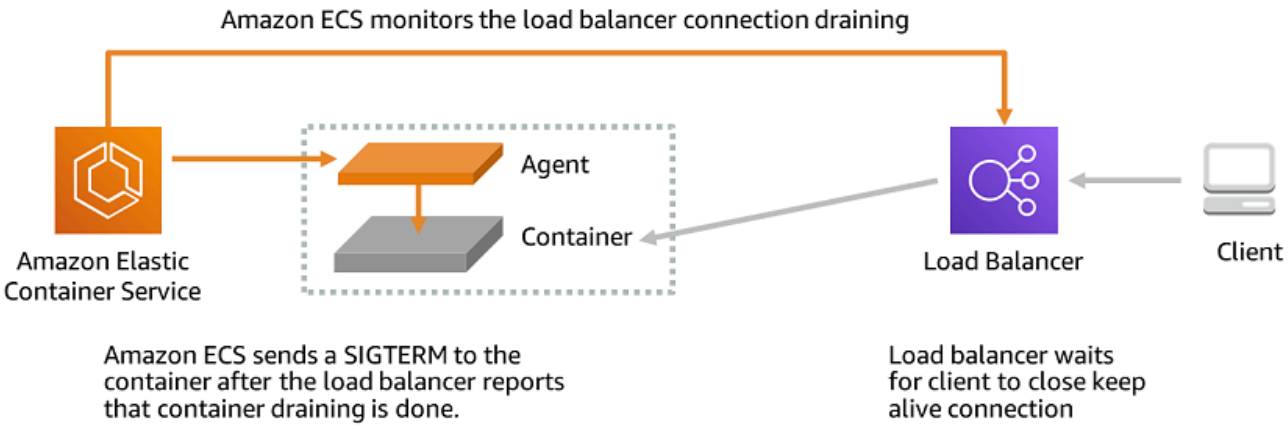
Avec ce paramètre, le processus de vérification de l'état prend 10 secondes, contre deux minutes et 30 secondes par défaut.

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.

Paramètres de vidange de la connexion de l'équilibreur de charge pour accélérer les délais de déploiement d'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. Lorsque vous demandez à l'équilibreur de charge d'arrêter le trafic vers le conteneur, il 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 une 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 :

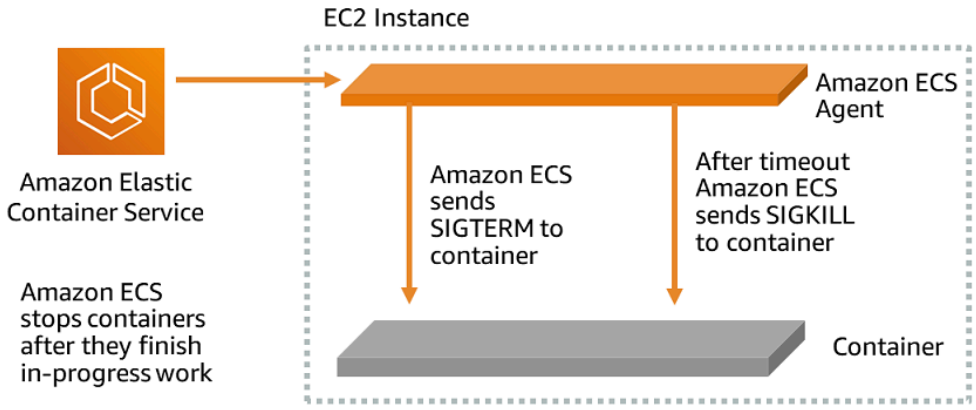
- `deregistration_delay.timeout_seconds` : 5

> **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 du 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.

Le schéma suivant montre comment Amazon ECS met fin à une tâche.



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 :

> **Note**
>
> Si votre application prend plus d'une seconde, multipliez la valeur par 2 et utilisez ce nombre comme valeur.

- `ECS_CONTAINER_STOP_TIMEOUT` : 2

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. En effet, sa boucle d'événements n'a plus 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.

Comportement d'extraction des images de conteneur pour des déploiements plus rapides d'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).

Types de lancement de Fargate

Fargate ne met pas en cache les images et, par conséquent, l'image entière est extraite du registre lorsqu'une tâche s'exécute. Nous recommandons ce qui suit pour les images utilisées pour les tâches Fargate :

- Utilisez une taille de tâche plus importante avec des vCPU supplémentaires.

  L'augmentation de la taille de la tâche permet de réduire le temps nécessaire à l'extraction de l'image lors du lancement d'une tâche.
- Utilisez une image de base plus petite.
- Conservez le référentiel qui stocke l'image dans la même région que la tâche.

Types de lancement d'AMI Windows sur Fargate

Les AMI Fargate Windows mettent en cache l'image de base du serveur du mois le plus récent et du mois précédent fournie par Microsoft. Ces images correspondent au numéro KB/build des patchs mis à jour chaque Patch Tuesday. Par exemple, le 8/8/2023, Microsoft a publié le KB5029247 (17763.4737) pour Windows Server 2019. Le mois précédent KB au 11/07/2023 était de 5028168 KB (17763.4645). Ainsi, pour les plateformes `WINDOWS_SERVER_2019_CORE` et `WINDOWS_SERVER_2019_FULL` les images de conteneurs suivantes ont été mises en cache :

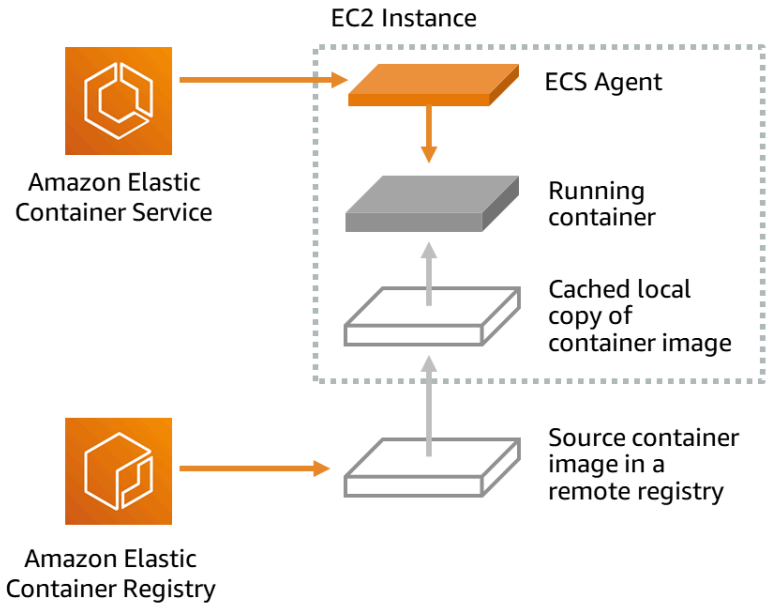
- `mcr.microsoft.com/windows/servercore:ltsc2019`
- `mcr.microsoft.com/windows/servercore:10.0.17763.4737`
- `mcr.microsoft.com/windows/servercore:10.0.17763.4645`

De plus, le 8/8/2023, Microsoft a publié le KB5029250 (20348.1906) pour Windows Server 2022. Le numéro de base du mois précédent au 11/07/2023 était KB5028171 (20348.1850). 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:10.0.20348.1906`
- `mcr.microsoft.com/windows/servercore:10.0.20348.1850`

Types de lancement d'Amazon EC2

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 `ECS_IMAGE_PULL_BEHAVIOR` agent détermine le comportement d'extraction de l'image et possède les valeurs suivantes :

- `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.

Paramètres du service Amazon ECS pour des déploiements Amazon ECS plus rapides

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 du service ECS que vous pouvez utiliser pour modifier le numéro :

- `minimumHealthyPercent`: 100 % (par défaut)

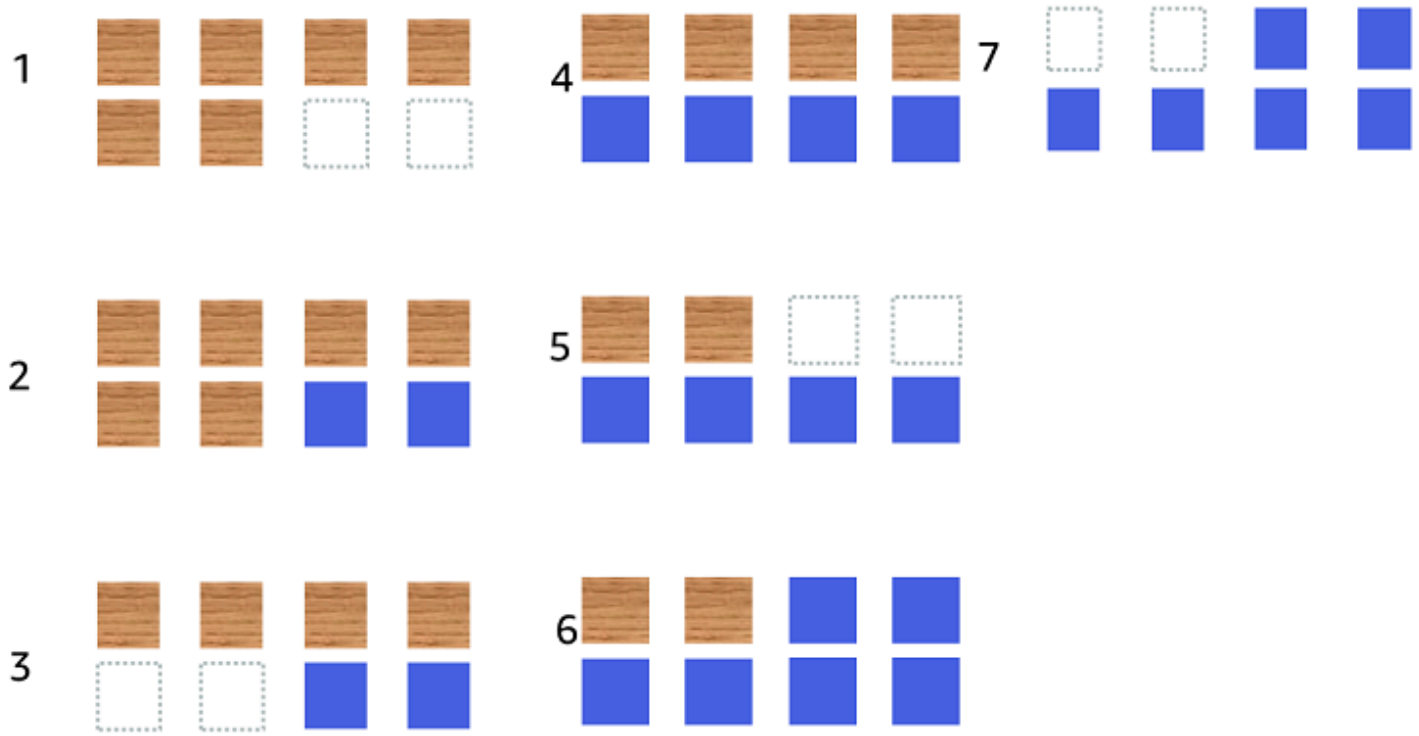
  Limite inférieure du nombre de tâches de votre service qui doivent rester `RUNNING` inchangées pendant un déploiement. Il s'agit d'un pourcentage de l'entier `desiredCount` arrondi à l'entier

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.

Prenons l'exemple du service suivant qui comporte six tâches tan, déployé dans un cluster pouvant accueillir huit tâches au total. Les options de configuration du service Amazon ECS par défaut n'autorisent pas le déploiement à moins de 100 % des six tâches souhaitées.



Le processus de déploiement est le suivant :

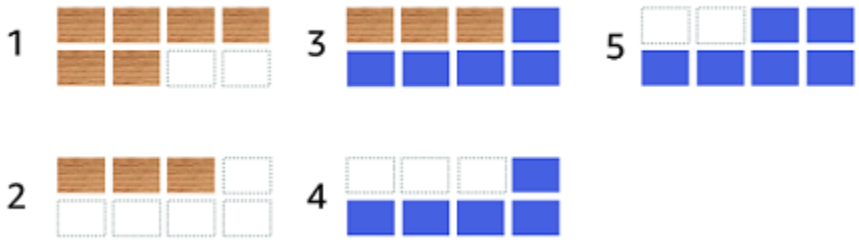
1. L'objectif est de remplacer les tâches en beige par des tâches bleues.
2. Le planificateur lance deux nouvelles tâches bleues car les paramètres par défaut exigent que six tâches soient en cours d'exécution.
3. Le planificateur arrête deux des tâches en bronzage car il y en aura un total de six (quatre en beige et deux en bleu).
4. Le planificateur lance deux tâches bleues supplémentaires.

5. Le planificateur arrête deux des tâches de bronzage.
6. Le planificateur lance deux tâches bleues supplémentaires.
7. Le planificateur arrête les deux dernières tâches de bronzage.

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.

Vous pouvez accélérer le déploiement en définissant la `minimumHealthyPercent` valeur sur 50 %.

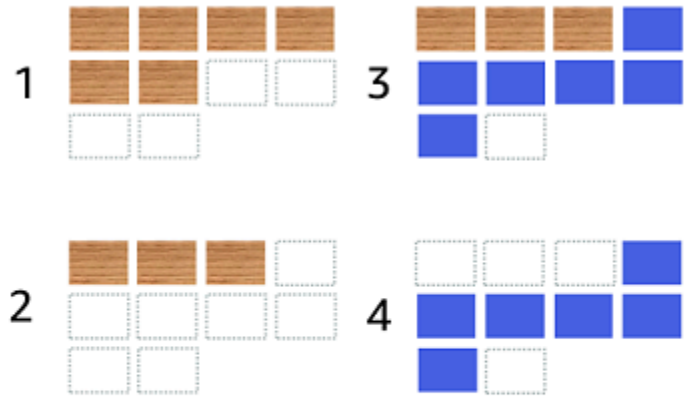
Prenons l'exemple du service suivant qui comporte six tâches tan, déployé dans un cluster pouvant accueillir huit tâches au total.



Le processus de déploiement est le suivant :

1. L'objectif est de remplacer les tâches en beige par des tâches bleues.
2. Le planificateur arrête trois des tâches de bronzage. Trois tâches de bronzage sont toujours en cours d'exécution, ce qui correspond à la `minimumHealthyPercent` valeur.
3. Le planificateur lance cinq tâches bleues.
4. Le planificateur arrête les trois tâches restantes.
5. Le planificateur lance les dernières tâches bleues.

Vous pouvez également ajouter de l'espace libre afin de pouvoir exécuter des tâches supplémentaires.



Le processus de déploiement est le suivant :

1. L'objectif est de remplacer les tâches en beige par des tâches bleues.
2. Le planificateur arrête trois des tâches de bronzage
3. Le planificateur lance six tâches bleues
4. Le planificateur arrête les trois tâches de bronzage.

Utilisez les valeurs suivantes pour les options de configuration du service Amazon ECS 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 service à l'aide de la console

Vous pouvez créer un service à l'aide de la console.

Tenez compte des éléments suivants lorsque vous utilisez la console :

- 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 vos tâches directement sur Fargate ou sur les instances Amazon EC2 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 des tâches par défaut répartit les tâches de manière uniforme 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 plus d’informations, consultez Disjoncteur de déploiement.
- Si vous souhaitez utiliser l'option de déploiement bleu/vert, déterminez le mode de CodeDeploy déplacement des applications. Les options suivantes sont disponibles :
  - CodeDeployDefault.ECS AllAtOnce : transfère immédiatement tout le trafic vers le conteneur Amazon ECS mis à jour
  - CodeDeployDefault.ECSlinear10 PercentEvery 1Minutes : déplace 10 % du trafic chaque minute jusqu'à ce que tout le trafic soit transféré.

  - CodeDeployDefault.ECSlinear10 PercentEvery 3Minutes : déplace 10 % du trafic toutes les 3 minutes jusqu'à ce que tout le trafic soit déplacé.
  - CodeDeployDefault.ecscanary10percent5minutes : déplace 10 % du trafic dès le premier incrément. Les 90 % restants sont déployés 5 minutes plus tard.
  - CodeDeployDefault.ecscanary10percent15minutes : déplace 10 % du trafic dès le premier incrément. Les 90 % restants sont déployés 15 minutes plus tard.
- 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 plus d'informations, consultez Interconnexion des services.
- Vous devez utiliser AWS CloudFormation ou AWS Command Line Interface déployer un service qui utilise l'un des paramètres suivants :
  - Stratégie de suivi avec une métrique personnalisée
  - Service de mise à jour : vous ne pouvez pas mettre à jour la configuration du `awsvpc` réseau ni le délai de grâce du bilan de santé.

  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, consultez AWS::ECS::Servicele guide de AWS CloudFormation l'utilisateur.

Créer rapidement un service

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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).
5. Sous Deployment configuration (Configuration du déploiement), spécifiez la manière dont votre application est déployée.

   a. Pour Application type (Type d'application), choisissez Service.

   b. Pour Task definition (Définition de tâche), choisissez la famille et la révision de définition de tâche à utiliser.

   c. Pour Service name (Nom du service), saisissez un nom pour votre service.

   d. Pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer et à conserver dans le service.
6. (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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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).
Type de lancement	a. Sur la page Définitions de tâches, sélectionnez le bouton d'option situé à côté de la définition de tâche. b. Dans le menu Déployer, choisissez Create service.

3. (Facultatif) Choisissez comment vos tâches sont distribuées dans votre infrastructure de cluster. Développez Compute configuration (Configuration de calcul), puis choisissez votre option.

Méthode de distribution	Étapes
Stratégie de fournisseur de capacité	a. Sous Options de calcul, choisissez Stratégie de fournisseur de capacité. b. Choisissez une stratégie :

Méthode de distribution	Étapes
	• 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. **Note** Pour utiliser un fournisseur de capacité dans une stratégie, le fournisseur de capacité doit être associé au cluster. Pour de plus

Méthode de distribution	Étapes
	amples informations sur les stratégies de fournisseurs de capacité, veuillez consulter Fournisseurs de capacité Amazon ECS.
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.

4. Pour spécifier le mode de déploiement de votre service, accédez à la section Configuration du déploiement, puis choisissez vos options.

   a. Pour Type d'application, laissez le choix sur Service.

   b. Pour Task definition (Définition de tâche) et Revision (Révision), choisissez la famille et la révision de définition de tâche à utiliser.

   c. Pour Service name (Nom du service), saisissez un nom pour votre service.

   d. 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.

Pour plus d'informations, consultez the section called "Concepts du planificateur de service".

e. Si vous choisissez Replica pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer et à conserver dans le service.

f. 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.	

g. 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.

5. (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.

   c. (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 du conteneur à CloudWatch Logs. Les autres options du pilote de journal sont configurées à l'aide de AWS FireLens. Pour plus d'informations, consultez Utilisation du routage personnalisé des journaux.

      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. Pour plus d'informations, consultez Transférer des journaux vers un domaine Amazon OpenSearch Service.
- 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.

6. (Facultatif) Pour utiliser la découverte de services, sélectionnez Utiliser la découverte de services, puis spécifiez les éléments suivants.

   a. 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.

   b. Fournissez des informations sur le service Service Discovery, telles que le nom et la description du service.

   c. 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.

   d. 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.

7. (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), sélectionnez 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 Load balancer name (nom d'équilibreur de charge), saisissez un nom unique. d. Pour Choose container to load balance (Choisissez le conteneur pour équilibrer la charge), choisissez le

Pour utiliser cet équilibreur de charge	Faites ceci	
	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. 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 surveilla	

Pour utiliser cet équilibreur de charge	Faites ceci	
	nce de l'état), saisissez un chemin qui existe dans votre conteneur où Application Load Balancer doit envoyer périodiqu ement des requêtes pour vérifier l'état de la connexion entre Applicati on Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (/). i. Pour Health check grace period (Période de grâce de surveillance de l'état), saisissez la durée (en secondes) pendant laquelle le planificateur de service doit ignorer les surveillances de l'état de la cible Elastic Load Balancing non saines.	

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 équilibrer 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 désenregistrement, entrez le nombre de secondes pendant lequel l'équilibreur 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 (/). g. Pour Health check grace period (Période de grâce de surveillance de l'état), saisissez la durée (en secondes) pendant laquelle le planificateur de service doit ignorer les surveillances de l'état de la cible Elastic Load Balancing non saines.	

8. (Facultatif) Pour configurer le service Auto Scaling, développez Service auto scaling, puis spécifiez les paramètres suivants.

   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 stratégie…	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. • ECS ServiceAverage CPUUtilization : utilisation moyenne du processeur du service. • ECS ServiceAverageMemoryUtilization — Utilisation moyenne de la mémoire du service. • ALB RequestCountPerTarget — Nombre de demandes traitées par cible dans un groupe cible Application Load Balancer. d. PourTarget value (Valeur cible), entrez la valeur conservée par le	

Pour utiliser ce type de stratégie…	Faites ceci...	
	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). h. • (Facultatif) Sélectionnez Désactiver la mise à l'échelle si vous	

Pour utiliser ce type de stratégie…	Faites ceci...	
	souhaitez que votre politique de dimension 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 stratégie…	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 stratégie…	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 stratégie…	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 stratégie…	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.	

9. (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 plus d'informations, consultez 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.

10. 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.

    > **Important**
    >
    > Seuls les sous-réseaux privés sont pris en charge pour le mode réseau `awsvpc`. Les tâches ne reçoivent pas les adresses IP publiques. Par conséquent, une passerelle NAT est requise pour l'accès Internet sortant. Par ailleurs, le trafic Internet entrant est acheminé par le biais d'un équilibreur de charge.

    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.

11. 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 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'opérations d'entrée/sortie (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.

Pour configurer ce type de volume	Faites ceci	
	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 en saisissant un identifiant de capture d'écran. f. 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	

Pour configurer ce type de volume	Faites ceci	
	système de fichiers ne correspond pas, la tâche ne démarrera pas. g. 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 comme guide pour créer et associer votre propre politique avec des autorisations répondant à vos besoins spécifiqu es. Pour plus d'informa tions sur les autorisations nécessaires, consultez Rôle IAM dans l'infrast ructure Amazon ECS. h. Pour le chiffrement, choisissez Par défaut si vous souhaitez utiliser le chiffrement Amazon EBS par défaut. Si le chiffrement est configuré	

Pour configurer ce type de volume	Faites ceci	
	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. i. 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	

Pour configurer ce type de volume	Faites ceci	
	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. j. (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 balises depuis le service, choisissez Service for Propagate tags	

Pour configurer ce type de volume	Faites ceci	
	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.	

12. (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.

Mise à jour d'un service à l'aide de la console

Vous pouvez mettre à jour un service Amazon ECS à l'aide de la console Amazon ECS. La configuration de service actuelle est préremplie. Vous pouvez modifier la définition de tâche, le nombre de tâches souhaité, la stratégie de fournisseur de capacité, la version de plateforme et la configuration du déploiement, ou toute combinaison de ces éléments.

Pour plus d'informations sur la mise à jour de la configuration de déploiement bleu/vert, veuillez consulter Mise à jour d'une configuration de déploiement bleu/vert à l'aide de la console.

Tenez compte des éléments suivants lorsque vous utilisez la console :

Si vous souhaitez arrêter temporairement votre service, définissez les tâches souhaitées sur 0. Ensuite, lorsque vous êtes prêt à démarrer le service, mettez-le à jour avec le nombre de tâches souhaitées d'origine.

Tenez compte des éléments suivants lorsque vous utilisez la console :

- Vous devez utiliser le AWS Command Line Interface pour mettre à jour un service qui utilise l'un des paramètres suivants :
  - Déploiements bleu/vert
  - Service Discovery : vous pouvez uniquement consulter votre configuration Service Discovery.
  - Stratégie de suivi avec une métrique personnalisée
  - Service de mise à jour : vous ne pouvez pas mettre à jour la configuration du `awsvpc` réseau ni le délai de grâce du bilan de santé.

  Pour plus d'informations sur la mise à jour d'un service à l'aide du AWS CLI, reportez-vous update-serviceà la section AWS Command Line Interface Référence.
- Si vous modifiez les ports utilisés par les conteneurs d'une définition de tâche, vous devrez mettre à jour vos groupes de sécurité d'instances de conteneur afin de fonctionner avec les ports mis à jour.
- 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.
- Si votre service utilise un équilibreur de charge, il n'est pas possible de modifier avec la console la configuration de ce dernier définie pour votre service lorsqu'il a été créé. Vous pouvez plutôt

utiliser le SDK AWS CLI ou 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 pouvez mettre à jour un service en cours d'exécution pour modifier certains des paramètres de configuration du service, tels que le nombre de tâches gérées par un service, la définition de tâche utilisée par les tâches ou, si vos tâches utilisent le type de lancement Fargate, vous pouvez modifier la version de plateforme utilisée par votre service. 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. Si votre application a besoin de plus de capacité, vous pouvez mettre à l'échelle votre service. Si vous n'utilisez pas toute votre capacité, vous pouvez la réduire, vous pouvez diminuer le nombre de tâches souhaitées dans votre service et libérer des ressources.

Si vous souhaitez utiliser une image de conteneur mise à jour pour vos tâches, vous pouvez créer une nouvelle révision de définition de tâche avec cette image et la déployer sur votre service à l'aide de l'option force new deployment (forcer le nouveau déploiement) dans la console.

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 Mise à jour propagée (`ECS`), le pourcentage minimum d'instances saines représente la limite inférieure pour le nombre de tâches de votre 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). Le paramètre s'applique également à toutes les instances de conteneur ayant l'état `DRAINING` si le service contient des tâches utilisant le type de lancement EC2. 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. Les tâches des services qui n'utilisent pas d'équilibreur de charge sont considérées comme saines si elles ont l'état `RUNNING`. 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. La valeur par défaut pour le pourcentage minimum d'instances saines est 100 pour cent.

Si un service utilise le type de déploiement Mise à jour propagée (ECS), le paramètre pourcentage maximum représente une limite supérieure du nombre de tâches dans un service autorisées à avoir l'état PENDING, RUNNING ou STOPPING pendant un déploiement, en tant que pourcentage du nombre de tâches souhaité (arrondi à l'entier le plus proche). Le paramètre s'applique également à toutes les instances de conteneur ayant l'état DRAINING si le service contient des tâches utilisant le type de lancement EC2. 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. 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. Si l'état de santé de la tâche de remplacement est 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).

> **Note**
>
> Ce comportement ne s'applique pas aux tâches exécutées et gérées par les services qui utilisent la mise à jour propagée comme type de déploiement. Lors d'une mise à jour propagée, le planificateur de service arrête d'abord les tâches défectueuses, puis lance les tâches de remplacement.

> **Important**
>
> Si vous modifiez les ports utilisés par les conteneurs d'une définition de tâche, vous devez mettre à jour vos groupes de sécurité d'instances de conteneur afin de fonctionner 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 lors de la création du service doivent rester dans la définition de tâche.
> 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.

Pour mettre à jour un service (console Amazon ECS)

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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. Pour Tâches souhaitées, entrez le nombre de tâches que vous souhaitez exécuter pour le service.
7. 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).
8. 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).
9. 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.
10. 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.

11. (Facultatif) Pour configurer le service Auto Scaling, développez Service auto scaling, puis spécifiez les paramètres suivants.

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 stratégie…	Faites ceci...	
Suivi de la cible	a. Pour Scaling policy type (Type de politique de mise à l'échelle	

Pour utiliser ce type de stratégie…	Faites ceci...	
	), 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. • ECS ServiceAverage CPUUtilization : utilisation moyenne du processeur du service. • ECS ServiceAverageMemoryUtilization — Utilisation moyenne de la mémoire du service. • ALB RequestCountPerTarget — Nombre de demandes traitées par cible dans un groupe cible Application Load Balancer. 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,	

Pour utiliser ce type de stratégie…	Faites ceci...	
	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 dimensionnement soit adaptée à l'augmentation du trafic, mais que vous	

Pour utiliser ce type de stratégie…	Faites ceci...	
	n'avez pas besoin qu'elle s'adapte lorsque le trafic diminue.	

Pour utiliser ce type de stratégie…	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 stratégie…	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 stratégie…	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 stratégie…	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.	

12. (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 à d'autres services de l'espace de noms, sélectionnez Client side only (Côté client uniquement).
    - 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.

13. 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. c. Pour les IOPS, entrez le nombre maximum d'opérations d'entrée/sortie (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.

Pour configurer ce type de volume	Faites ceci	
	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 en saisissant un identifiant de capture d'écran. f. 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	

Pour configurer ce type de volume	Faites ceci	
	système de fichiers ne correspond pas, la tâche ne démarrera pas. g. 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 comme guide pour créer et associer votre propre politique avec des autorisations répondant à vos besoins spécifiqu es. Pour plus d'informa tions sur les autorisations nécessaires, consultez Rôle IAM dans l'infrast ructure Amazon ECS. h. Pour le chiffrement, choisissez Par défaut si vous souhaitez utiliser le chiffrement Amazon EBS par défaut. Si le chiffrement est configuré	

Pour configurer ce type de volume	Faites ceci	
	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. i. 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	

Pour configurer ce type de volume	Faites ceci	
	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. j. (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 balises depuis le service, choisissez Service for Propagate tags	

Pour configurer ce type de volume	Faites ceci	
	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.	

14. (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.

15. Choisissez Mettre à jour.

Mise à jour d'une configuration de déploiement bleu/vert à l'aide de la console

Vous pouvez mettre à jour une configuration de déploiement bleu/vert à l'aide de la console Amazon ECS. La configuration de déploiement bleu/vert actuelle est préremplie. Vous pouvez mettre à jour les options de déploiement bleu/vert suivantes :

- Nom du groupe de déploiement : paramètres CodeDeploy de déploiement
- Nom de l'application : groupe CodeDeploy de déploiement
- Configuration du déploiement : comment CodeDeploy achemine le trafic de production vers votre ensemble de tâches de remplacement lors d'un déploiement
- Écouteur de test sur l'équilibreur de charge : CodeDeploy utilise l'écouteur de test pour acheminer votre trafic de test vers l'ensemble de tâches de remplacement lors d'un déploiement

Vous devez configurer la nouvelle option avant de mettre à jour la configuration.

Pour mettre à jour une configuration de déploiement bleu/vert (console Amazon ECS)

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, sélectionnez le cluster.
3. Sur la page Cluster overview (Aperçu du cluster), sélectionnez le service, puis choisissez Update (Mettre à jour).
4. Développez les options de déploiement - Propulsé par CodeDeploy, puis choisissez les options à mettre à jour :
   - Pour modifier le groupe CodeDeploy de déploiement, dans Nom de l'application, choisissez le groupe de déploiement.
   - Pour modifier les paramètres de CodeDeploy déploiement, choisissez le groupe dans Nom du groupe de déploiement.
   - Pour modifier la manière dont le trafic de production est CodeDeploy acheminé vers votre ensemble de tâches de remplacement lors d'un déploiement, sélectionnez l'option dans Configuration du déploiement.
5. Sélectionnez les hooks d'événement de cycle de vie de déploiement et les fonctions Lambda associées à exécuter dans le cadre de la nouvelle révision du service de déploiement. Les hooks de cycle de vie disponibles sont les suivants :

- BeforeInstall— Utilisez ce hook d'événements du cycle de vie de déploiement pour appeler une fonction Lambda avant la création de l'ensemble de tâches de remplacement. Le résultat de la fonction Lambda pour cet événement du cycle de vie ne lance aucune restauration.
- AfterInstall— Utilisez ce hook d'événements du cycle de vie de déploiement pour appeler une fonction Lambda après la création de l'ensemble de tâches de remplacement. Le résultat de la fonction Lambda pour cet événement du cycle de vie peut lancer une restauration.
- BeforeAllowTraffic— Utilisez ce hook d'événements du cycle de vie du déploiement pour appeler une fonction Lambda avant que le trafic de production ne soit redirigé vers l'ensemble de tâches de remplacement. Le résultat de la fonction Lambda pour cet événement du cycle de vie peut lancer une restauration.
- AfterAllowTraffic— Utilisez ce hook d'événements du cycle de vie du déploiement pour appeler une fonction Lambda une fois que le trafic de production a été redirigé vers l'ensemble de tâches de remplacement. Le résultat de la fonction Lambda pour cet événement du cycle de vie peut lancer une restauration.

6. Pour modifier l'écouteur de test, développez l'équilibrage de charge, puis pour l'écouteur de test pour le CodeDeploy déploiement, choisissez l'écouteur de test.
7. Choisissez Mettre à jour.

Suppression d'un service avec la console

Vous pouvez supprimer un service Amazon ECS service à l'aide de la console. 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, 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.

Pour supprimer un service (console Amazon ECS)

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, sélectionnez le cluster pour le service.
3. Sur la page Clusters, choisissez le cluster.
4. Sur la page Cluster : ***name*** (Cluster : nom), 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.

Types de déploiement Amazon ECS

Un type de déploiement Amazon ECS détermine la stratégie de déploiement utilisée par votre service. Il existe trois types de déploiement : mise à jour propagée, bleu/vert et externe.

Vous pouvez afficher des informations sur le type de déploiement de services sur la page détails du service ou en utilisant l'API `describe-services`. Pour plus d'informations, consultez le DescribeServicesmanuel Amazon Elastic Container Service API Reference.

Rubriques

- La mise à jour propagée
- Déploiement bleu/vert avec CodeDeploy
- Utiliser un contrôleur tiers pour les déploiements externes d'Amazon ECS

La mise à jour propagée

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. La configuration de déploiement se compose des éléments suivants :

- `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.
- `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 plus d'informations, consultez Le service (*`nom-service`*) n'a pas pu arrêter ou démarrer des tâches lors d'un déploiement en raison de la configuration du déploiement du service. Mettez à jour la valeur minimumHealthyPercent ou MaximumPercent et réessayez..

Un déploiement propagé utilise le disjoncteur du circuit de déploiement pour déterminer si les tâches atteignent un état stable. Le disjoncteur du circuit de déploiement peut éventuellement restaurer un déploiement en cas de défaillance.

Méthodes de détection des échecs

Il existe deux méthodes permettant d'identifier rapidement l'échec d'un déploiement, puis de revenir éventuellement à l'échec jusqu'au dernier déploiement fonctionnel.

- the section called “Disjoncteur de déploiement”
- the section called “CloudWatch alarmes”

Les méthodes peuvent être utilisées séparément ou conjointement. Lorsque vous utilisez les deux méthodes, le déploiement est défini comme un échec dès que les critères d'échec de l'une ou l'autre méthode 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.

Pour plus d'informations sur les bonnes pratiques du processus de déploiement d'Amazon ECS, veuillez consulter Déploiement des tâches dans le Guide des bonnes pratiques Amazon ECS (langue française non garantie).

Disjoncteur de déploiement

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 plus d’informations, consultez 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 la section Création d'une EventBridge règle 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 restauration tant qu'il n'a pas atteint l'état `COMPLETED`. 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 assure le suivi des 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. EventBridgegénère 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 plus d'informations, consultez 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).

CloudWatch alarmes

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.

- 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 plus d'informations, consultez 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 plus d'informations, consultez 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.

Déploiement bleu/vert avec CodeDeploy

Le type de déploiement bleu/vert utilise le modèle de déploiement bleu/vert 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 Présentation d' CodeDeploy dans le Guide de l'utilisateur AWS CodeDeploy .

Le trafic peut être modifié de trois manières lors d'un déploiement bleu/vert :

- 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 déploiement bleu/vert : CodeDeploy

CodeDeploy candidature

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.ECS AllAtOnce : transfère immédiatement tout le trafic vers le conteneur Amazon ECS mis à jour
- CodeDeployDefault.ECSlinear10 PercentEvery 1Minutes : déplace 10 % du trafic chaque minute jusqu'à ce que tout le trafic soit transféré.
- CodeDeployDefault.ECSlinear10 PercentEvery 3Minutes : déplace 10 % du trafic toutes les 3 minutes jusqu'à ce que tout le trafic soit déplacé.
- CodeDeployDefault.ecscanary10percent5minutes : déplace 10 % du trafic dès le premier incrément. Les 90 % restants sont déployés 5 minutes plus tard.
- CodeDeployDefault.ecscanary10percent15minutes : 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 relatives au déploiement bleu/vert

Tenez compte des éléments suivants lorsque vous utilisez le type de déploiement bleu/vert :

- Lors de la création initiale d'un service Amazon ECS service utilisant le déploiement bleu/vert, 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 déploiement bleu/vert et qu'un événement de mise à l'échelle se produit, le trafic continue d'être déplacé 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.
  - Si un service est en cours de déploiement bleu/vert et qu'un événement de mise à l'échelle se produit, le nombre de tâches souhaité peut être défini sur une valeur inattendue. Cela est dû au fait que la scalabilité automatique considère le nombre de tâches en cours comme la capacité actuelle, ce qui représente le double du nombre approprié de tâches utilisées dans le calcul du nombre de tâches souhaité.
- 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 pour le déploiement bleu/vert

Les déploiements bleu/vert d'Amazon ECS sont rendus possibles par une combinaison d'Amazon ECS et d'API. CodeDeploy Les utilisateurs doivent disposer des autorisations appropriées pour ces services avant de pouvoir utiliser les déploiements bleu/vert d'Amazon ECS dans AWS Management Console ou avec les AWS CLI SDK.

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 plus d'informations, consultez 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 les API Amazon ECS, 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 service utilisant le type de déploiement bleu/vert, vous devez créer un rôle IAM (`ecsCodeDeployRole`). Pour plus d'informations, consultez Rôle CodeDeploy IAM d'Amazon ECS.

Les exemples de politique IAM Exemple de création de service et Exemple de mise à jour du service présentent les autorisations nécessaires aux utilisateurs pour utiliser des déploiements bleus/verts Amazon ECS dans la AWS Management Console.

Utiliser un contrôleur tiers pour les déploiements externes d'Amazon ECS

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`.

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 : chaîne

   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. Pour plus d'informations, consultez deploymentConfiguration.

`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 plus d'informations, consultez Balisage des ressources Amazon ECS.

`key`

Type : chaîne

Contraintes de longueur : Longueur minimum de 1. Longueur maximum 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 : chaîne

Contraintes de longueur : Longueur minimum de 0. Longueur maximum 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 plus d'informations, consultez Identification de vos ressources pour facturation.

`propagateTags`

Type : chaîne

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`.

`healthCheckGracePeriodSeconds`

Type : entier

Obligatoire : non

Durée, en secondes, pendant laquelle le planificateur de service Amazon ECS service doit ignorer les surveillances de l'état des cibles Elastic Load Balancing non saines, les surveillances de l'état du conteneur et les surveillances de l'état de Route 53 après le passage d'une tâche à l'état `RUNNING`. Cela est valable uniquement si votre service est configuré pour utiliser un équilibreur de charge. Si votre service dispose d'un équilibreur de charge défini et que vous ne spécifiez pas de valeur de période de grâce de surveillance de l'état, c'est la valeur par défaut de `0` qui est utilisée.

Si des tâches de votre service prennent du temps pour démarrer et répondre aux surveillances de l'état, vous pouvez spécifier une période de grâce pouvant atteindre 2 147 483 647 secondes, pendant laquelle le planificateur de service ECS ignore le statut de la surveillance de l'état. Cette période de grâce peut empêcher le planificateur de service ECS de marquer les tâches comme étant non saines et de les arrêter avant qu'elles aient le temps de s'exécuter.

Si vous n'utilisez pas un Elastic Load Balancing, nous vous recommandons d'utiliser `startPeriod` dans les paramètres du contrôle de surveillance de l'état de la définition de la tâche. Pour plus d'informations, consultez Health check (Surveillance de l'état).

`schedulingStrategy`

Stratégie de planification à utiliser. Les services utilisant un contrôleur de déploiement externe prennent uniquement en charge la stratégie de planification `REPLICA`. Pour plus d'informations, consultez Concepts du planificateur de service.

`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": ""
        }
```

```
    ],
    "healthCheckGracePeriodSeconds": 0,
    "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 : chaîne

   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 plus d'informations, consultez 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 plus d'informations, consultez Versions de plateforme Linux Fargate.

> **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 Amazon EC2, consultez Fargate Task Networking. Pour plus d'informations sur la mise en réseau pour le type de lancement Fargate, consultez. Mise en réseau des tâches pour les tâches sur Fargate

`serviceRegistries`

Les détails des enregistrements de découverte de service à attribuer à ce service. Pour plus d'informations, consultez Découverte de service.

`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.

Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge

Vous pouvez configurer votre service Amazon ECS service de manière à utiliser Elastic Load Balancing pour répartir uniformément le trafic 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 types d'équilibreurs de charge Application Load Balancer et Network Load Balancer. Les équilibreurs de charge Application Load Balancer permettent d'acheminer le trafic HTTP/HTTPS (ou de couche 7). Les équilibreurs de charge Network Load Balancer permettent d'acheminer le trafic TCP ou UDP (ou couche 4).

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 sur des instances Fargate ou EC2.
- 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).

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. 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.

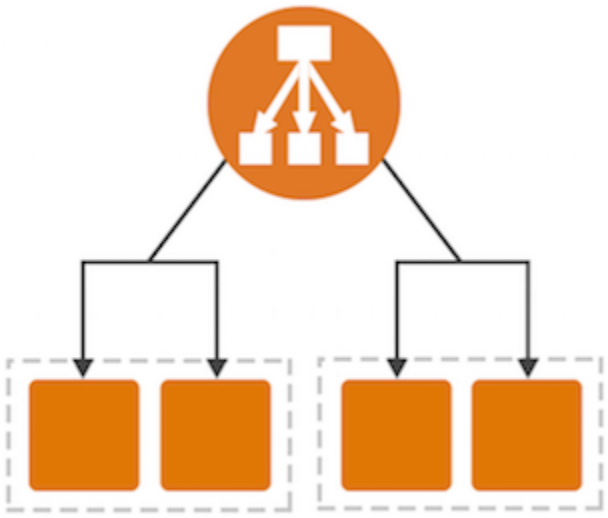
Avec votre équilibreur de charge, vous payez uniquement en fonction de votre utilisation. Pour plus d'informations, veuillez consultez Tarification Elastic Load Balancing.

Rubriques

- Utiliser un Application Load Balancer pour Amazon ECS
- Utiliser un Network Load Balancer pour Amazon ECS
- Enregistrement de plusieurs groupes cibles auprès d'un service

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 dans l'équilibreur de charge Application 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, consultez le Guide de l'utilisateur pour les équilibreurs de charge Application Load Balancer.



Considérations relatives à l'Application Load Balancer

Les considérations suivantes sont spécifiques aux services Amazon ECS service qui utilisent des équilibreurs de charge Application Load Balancer ou des Network Load Balancer :

- 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 plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS.
- 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.
- 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 réseau `awsvpc` sont associées à une interface réseau Elastic et non à une instance Amazon EC2.
- Si votre service utilise un équilibreur de charge Application Load Balancer et requiert un accès à plusieurs ports avec équilibrage de charge, comme les ports 80 et 443 pour un service HTTP/HTTPS, 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 façon 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 Network Load Balancer configuré avec des adresses IP comme cibles et que la préservation des adresses IP client est désactivée, les requêtes sont considérées comme provenant de l'adresse IP privée des Network Load Balancers. Cela signifie que les services associés à un Network Load Balancer sont effectivement ouverts au monde dès que vous autorisez les demandes entrantes et les bilans de santé du groupe de sécurité cible.
- Afin d'utiliser un Network Load Balancer pour acheminer le trafic UDP vers vos tâches Amazon ECS sur Fargate, vous devez avoir la version de plateforme `1.4.0` (Linux) ou `1.0.0` (Windows).
- Réduisez les erreurs dans vos applications clientes en définissant `StopTimeout`, dans la définition de la tâche, un délai d'annulation de l'enregistrement du groupe cible plus long que le délai d'expiration de la connexion de votre client. Pour plus d'infirmations sur la configuration client recommandée, consultez la bibliothèque Builders ici.

  De plus, l'attribut de groupe cible Network Load Balancer pour la résiliation de la connexion ferme toutes les connexions restantes après la date de l'annulation de l'inscription. Si le client ne les gère pas, ils peuvent alors afficher des messages d'erreur indésirables.
- Si vous rencontrez des problèmes avec l'équilibreur de charge utilisé par vos services, consultez Résolution des problèmes liés aux équilibreurs de charge des services.
- Vos tâches et votre équilibreur de charge (Application Load Balancer) ou Network Load Balancer) doivent se trouver dans le même VPC.

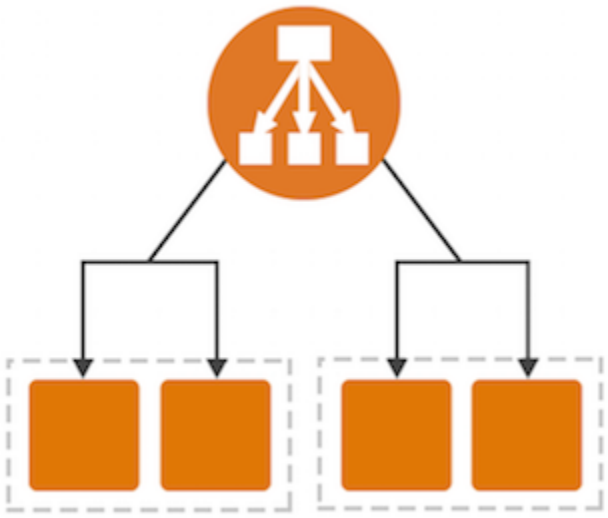
- La préservation des adresses IP client du Network Load Balancer est également compatible avec les cibles Fargate.
- Utilisez un groupe cible unique pour chaque service.

  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 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.



Considérations relatives au Network Load Balancer

Les considérations suivantes sont spécifiques aux services Amazon ECS service qui utilisent des équilibreurs de charge Application Load Balancer ou des Network Load Balancer :

- 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 plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS.
- 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.
- 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 réseau `awsvpc` sont associées à une interface réseau Elastic et non à une instance Amazon EC2.
- Si votre service utilise un équilibreur de charge Application Load Balancer et requiert un accès à plusieurs ports avec équilibrage de charge, comme les ports 80 et 443 pour un service HTTP/HTTPS, 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 façon 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 Network Load Balancer configuré avec des adresses IP comme cibles et que la préservation des adresses IP client est désactivée, les requêtes sont considérées comme provenant de l'adresse IP privée des Network Load Balancers. Cela signifie que les services associés à un Network Load Balancer sont effectivement ouverts au monde dès que vous autorisez les demandes entrantes et les bilans de santé du groupe de sécurité cible.
- Afin d'utiliser un Network Load Balancer pour acheminer le trafic UDP vers vos tâches Amazon ECS sur Fargate, vous devez avoir la version de plateforme `1.4.0` (Linux) ou `1.0.0` (Windows).
- Réduisez les erreurs dans vos applications clientes en définissant `StopTimeout`, dans la définition de la tâche, un délai d'annulation de l'enregistrement du groupe cible plus long que le délai d'expiration de la connexion de votre client. Pour plus d'infirmations sur la configuration client recommandée, consultez la bibliothèque Builders ici.

  De plus, l'attribut de groupe cible Network Load Balancer pour la résiliation de la connexion ferme toutes les connexions restantes après la date de l'annulation de l'inscription. Si le client ne les gère pas, ils peuvent alors afficher des messages d'erreur indésirables.
- Si vous rencontrez des problèmes avec l'équilibreur de charge utilisé par vos services, consultez Résolution des problèmes liés aux équilibreurs de charge des services.
- Vos tâches et votre équilibreur de charge (Application Load Balancer) ou Network Load Balancer) doivent se trouver dans le même VPC.

- La préservation des adresses IP client du Network Load Balancer est également 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 réseau `awsvpc` sont associées à une interface réseau Elastic et non à une instance Amazon EC2.
> Vous ne pouvez pas enregistrer des instances par ID d'instance si elles ont les types d'instance suivants : C1, CC1, CC2, CG1, GC2, CR1, G1, G2, HI1 et HS1, M1, M2, M3 et T1. Vous pouvez enregistrer les instances de ces types par adresse IP.

Enregistrement de plusieurs groupes cibles auprès d'un service

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 relatives aux groupes cibles multiples

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 groupes cibles multiples est prise en charge pour des services contenant des tâches qui utilisent à la fois les types de lancement Fargate et 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 plus d’informations, consultez 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.

Exemple : avec des 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
   }
]
```

Exemple : exposition de plusieurs ports à partir du 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
   }
]
```

Exemple : exposition de ports à partir 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

La scalabilité automatique est l'aptitude à augmenter ou à diminuer automatiquement le nombre souhaité de tâches dans votre service Amazon ECS service. 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 plus d’informations, consultez Utilisation des services. 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 :

- Faites évoluer votre service Amazon ECS à l'aide d'une valeur métrique cible – 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.
- Faites évoluer votre service Amazon ECS à l'aide d'incréments prédéfinis basés sur CloudWatch les alarmes – 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.
- Faites évoluer votre service Amazon ECS à l'aide d'un calendrier—Augmentez ou diminuez le nombre de tâches exécutées par votre service en fonction de la date et de l'heure.

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. Pour plus d'informations, consultez 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.

Bonnes pratiques pour le dimensionnement 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 est mis en œuvre 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 fournissez les seuils de mise à l'échelle, soit en définissant une cible métrique, appelée échelle de suivi des cibles, soit en spécifiant des seuils, appelés mise à l'échelle par étapes. Une fois qu'Application Auto Scaling est configurée, elle 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.

Une application doit être étendue si l'on prévoit que la demande sera supérieure à la capacité actuelle. À l'inverse, une application peut être étendue pour réduire les coûts lorsque les ressources dépassent la demande.

Identifier une métrique

Pour effectuer une mise à l'échelle efficace, il est essentiel d'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 les ressources sont maintenues stables, mais que la demande change, la valeur 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, créez des tableaux de bord pour les indicateurs d'utilisation de votre application. Ces indicateurs incluent l'utilisation du processeur, l'utilisation de la mémoire, les opérations d'E/S, la profondeur de la file d'attente d'E/S et le débit du réseau. Vous pouvez collecter ces statistiques à l'aide d'un service tel que Container Insights. Pour plus d'informations, consultez Surveillez les conteneurs Amazon ECS à l'aide de Container Insights. 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.

Commencez par une petite demande ou un taux d'insertion professionnelle. 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 application soient trop lents pour atteindre vos objectifs de niveau de service (SLO).

Lors du test de charge, 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 le premier à un niveau élevé ou atteint un pic puis fait planter 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 la charge ajoutée encore plus importante, 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'instance Amazon EC2. 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'instances Amazon EC2. 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. Lorsqu'une charge supplémentaire est ajoutée, 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, des recherches supplémentaires sont nécessaires pour trouver une métrique appropriée.

Modèles d'application et propriétés de mise à l'échelle courants

Des logiciels de toutes sortes sont exécutés 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 de procéder à 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 utiliser efficacement plusieurs processeurs pour ce faire. Chaque demande est traitée soit par un thread dédié avec une faible surcharge de mémoire, soit par une boucle d'événements asynchrone qui 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.

Ce type d'application convient à la mise à l'échelle automatique basée sur le processeur. L'application bénéficie d'une flexibilité maximale en termes de mise à l'échelle. Il peut être redimensionné verticalement en lui fournissant des instances Amazon EC2 plus grandes ou des vCPU Fargate. De plus, il peut également être redimensionné 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 la capacité Amazon EC2 pour cette application, pensez à la placer sur des instances optimisées pour le calcul, telles que la famille or. `c5` `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.

Ce type d'application convient à la mise à l'échelle automatique basée sur la mémoire. L'application bénéficie d'une flexibilité maximale en termes de mise à l'échelle. Il peut être redimensionné à la fois verticalement en lui fournissant des ressources de mémoire Amazon EC2 ou Fargate plus importantes. De plus, il peut également être redimensionné 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 capacité Amazon EC2 pour cette application, pensez à la placer sur des instances optimisées pour la mémoire, telles que la famille or. `r5` `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, il y a toujours une simultanéité maximale que l'application peut prendre en charge. Généralement, la limite de simultanéité est définie proportionnellement aux ressources de mémoire disponibles. Si la limite de simultanéité est atteinte, des demandes supplémentaires sont placées dans une file d'attente de backlog. Si la file d'attente du backlog est débordée, les demandes entrantes supplémentaires sont immédiatement rejetées. 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. Étant donné qu'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. Cette métrique est calculée 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. La `ActiveConnectionCount` métrique doit être divisée par le nombre de répliques pour obtenir une valeur moyenne. La valeur moyenne doit être utilisée 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 en 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. Au lieu de cela, nous recommandons généralement 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.

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. 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.

Faites évoluer votre service Amazon ECS à l'aide d'une valeur métrique cible

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.

Considérations

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. Pour plus d'informations, consultez 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 métrique `ALBRequestCountPerTarget` pour les politiques de suivi des objectifs et d'échelonnement n'est pas prise en charge pour le type de déploiement bleu/vert.

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.

Pour configurer les politiques de dimensionnement des cibles pour votre service Amazon ECS à l'aide de la console Amazon ECS

1. Outre les autorisations IAM standard pour créer et mettre à jour des services, vous avez besoin d'autorisations supplémentaires. Pour plus d'informations, consultez Autorisations IAM requises pour Service Auto Scaling.
2. Vous pouvez configurer une politique de dimensionnement lorsque vous créez ou mettez à jour un service. Pour plus d'informations, consultez les étapes suivantes :
   - Créer un service à l'aide de paramètres définis— Crée un nouveau service
   - Mise à jour d'un service à l'aide de la console— Mettre à jour un service existant

Pour configurer les politiques de dimensionnement des cibles pour votre service Amazon ECS à l'aide du AWS CLI

1. Outre les autorisations IAM standard pour créer et mettre à jour des services, vous avez besoin d'autorisations supplémentaires. Pour plus d'informations, consultez Autorisations IAM requises pour Service Auto Scaling.
2. Enregistrez votre service Amazon ECS en tant que cible évolutive à l'aide de la register-scalable-targetcommande.
3. Créez une politique de dimensionnement à l'aide de la put-scaling-policycommande.

Faites évoluer votre service Amazon ECS à l'aide d'incréments prédéfinis basés sur CloudWatch les alarmes

Avec les politiques de dimensionnement par étapes, vous spécifiez les CloudWatch alarmes qui déclenchent le processus de dimensionnement. Par exemple, si vous souhaitez augmenter votre capacité lorsque l'utilisation du processeur atteint un certain niveau, créez une alarme à l'aide de la `CPUUtilization` métrique fournie. Lorsque vous créez une politique de dimensionnement d'étape, vous devez indiquer l'un des types d'ajustement suivants :

- Ajouter — Augmentez le nombre de tâches d'un nombre spécifié d'unités de capacité ou d'un pourcentage spécifié de la capacité actuelle.
- Supprimer — Diminuez le nombre de tâches d'un nombre spécifié d'unités de capacité ou d'un pourcentage spécifié de la capacité actuelle.
- Régler sur : définissez le nombre de tâches sur le nombre d'unités de capacité spécifié.

Par exemple, supposons que la capacité cible et la capacité fournie sont égales à 10 et que la politique de dimensionnement ajoute 1. Lorsque l'alarme est violée, le processus de dimensionnement automatique ajoute 1 à 10 pour obtenir 11. Amazon ECS lance donc une tâche pour le service.

Nous vous recommandons vivement d'utiliser des politiques de dimensionnement du suivi des cibles afin de vous adapter à des indicateurs tels 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 que Service Auto Scaling suit de près la courbe de demande de vos applications.

Pour une présentation des politiques de dimensionnement par étapes et de leur fonctionnement, consultez la section Politiques de dimensionnement par étapes du Guide de l'utilisateur d'Application Auto Scaling. Après avoir lu cette introduction, consultez les sections suivantes pour savoir comment configurer le dimensionnement par étapes pour Amazon ECS à l'aide de la console et AWS Command Line Interface.

Pour configurer les politiques de dimensionnement par étapes pour votre service Amazon ECS à l'aide de la console Amazon ECS

1. Outre les autorisations IAM standard pour créer et mettre à jour des services, vous avez besoin d'autorisations supplémentaires. Pour plus d'informations, consultez Autorisations IAM requises pour Service Auto Scaling.
2. Vous pouvez configurer une politique de dimensionnement lorsque vous créez ou mettez à jour un service. Pour plus d'informations, consultez les étapes suivantes :
   - Créer un service à l'aide de paramètres définis— Crée un nouveau service
   - Mise à jour d'un service à l'aide de la console— Mettre à jour un service existant

Pour configurer les politiques de dimensionnement par étapes pour votre service Amazon ECS à l'aide du AWS CLI

1. Outre les autorisations IAM standard pour créer et mettre à jour des services, vous avez besoin d'autorisations supplémentaires. Pour plus d'informations, consultez Autorisations IAM requises pour Service Auto Scaling.
2. Enregistrez votre service Amazon ECS en tant que cible évolutive à l'aide de la register-scalable-targetcommande.
3. Créez une politique de dimensionnement à l'aide de la put-scaling-policycommande.
4. Créez une alarme qui initie la politique de dimensionnement à l'aide de la put-metric-alarmcommande.

Faites évoluer votre service Amazon ECS à l'aide d'un calendrier

Avec la mise à l'échelle planifiée, vous pouvez configurer la mise à l'échelle automatique de votre application en fonction des changements de charge prévisibles en créant des actions planifiées qui augmentent ou diminuent la capacité 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'une capacité suffisante pour gérer le pic de trafic en milieu de semaine, mais elle ne surprovisionne pas de capacité inutile 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 mise à l'échelle planifiée, la stratégie de mise à l'échelle peut continuer à prendre des décisions sur l'opportunité de poursuivre la mise à l'échelle de la capacité. Cela vous permet de vous assurer que vous avez une capacité suffisante pour gérer la charge de votre application. Bien que votre application soit mise à l'échelle pour répondre à la demande, la capacité actuelle doit se situer dans les limites de la capacité minimale et maximale qui a été fixée 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.

Interconnexion des services

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 "Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge".

Choix d'une méthode d'interconnexion

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 les VPC du même Région AWS cluster.

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, 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.

Pour plus d’informations, consultez Service Connect.

- 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 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.

Pour plus d’informations, consultez Découverte de service.

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	oui	oui, mais nécessite que les clients soient

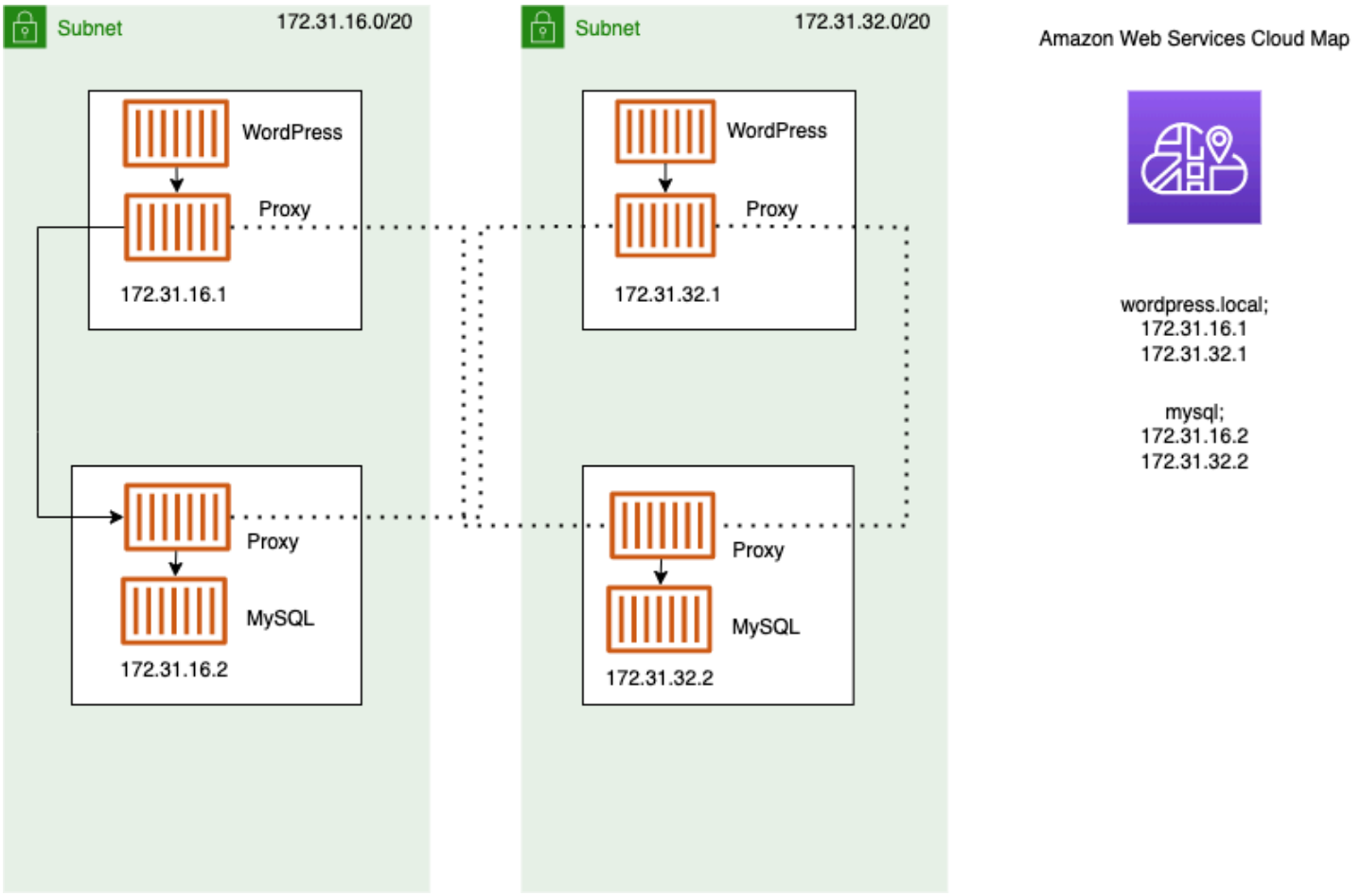
Options d'interconnexion	Pontée	**awsvpc**	Host (Hôte)
	au courant des enregistrements SRV dans le DNS sans `hostPort`.		au courant des enregistrements SRV dans le DNS sans `hostPort`.
Service Connect	oui	oui	non

Service Connect

Amazon ECS Service Connect permet de gérer les service-to-service communications en tant que configuration Amazon ECS. Pour ce faire, il crée à la fois une découverte de services et un maillage de services dans Amazon ECS. Cela fournit la configuration complète au sein de chaque service Amazon ECS que vous gérez par le biais de 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 Amazon VPC, ainsi que des métriques et des journaux standardisés pour surveiller toutes vos applications sur Amazon ECS. Amazon ECS Service Connect interconnecte uniquement les services Amazon ECS.

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.



Vue d'ensemble des étapes de configuration de Service Connect

Suivez ces étapes pour configurer Service Connect pour un groupe de services connexes.

> **Important**
>
> - Amazon ECS Service Connect crée AWS Cloud Map des services dans votre compte. La modification de ces AWS Cloud Map ressources en enregistrant/désenregistrant manuellement des instances, en modifiant les attributs des instances ou en supprimant un service peut entraîner un comportement inattendu pour le trafic de votre application ou pour les déploiements ultérieurs.
> - Amazon ECS 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 ECS avec un espace de AWS Cloud Map noms ou créez l'espace de noms séparément. Pour simplifier l'organisation, créez un cluster Amazon ECS avec 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. Amazon ECS 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.

Régions dans lesquelles Service Connect est disponible

Amazon ECS Service Connect est disponible dans les AWS régions suivantes :

Nom de la région	Région
USA Est (Ohio)	us-east-2
USA Est (Virginie du Nord)	us-east-1

Nom de la région	Région
US West (N. California)	us-west-1
US West (Oregon)	us-west-2
Afrique (Le Cap)	af-south-1
Asie-Pacifique (Hong Kong)	ap-east-1
Asie-Pacifique (Jakarta)	ap-southeast-3
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
Asie-Pacifique (Sydney)	ap-southeast-2
Asie-Pacifique (Melbourne)	ap-southeast-4
Asie-Pacifique (Tokyo)	ap-northeast-1
Canada (Central)	ca-central-1
Canada Ouest (Calgary)	ca-west-1
Chine (Beijing)	cn-north-1 (Remarque : le protocole TLS pour Service Connect n'est pas disponible dans cette région.)
Chine (Ningxia)	cn-northwest-1 (Remarque : le protocole TLS pour Service Connect n'est pas disponible dans cette région.)
Europe (Francfort)	eu-central-1

Nom de la région	Région
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
Europe (Zurich)	eu-central-2
Israël (Tel Aviv)	il-central-1
Moyen-Orient (Bahreïn)	me-south-1
Moyen-Orient (EAU)	me-central-1
Amérique du Sud (São Paulo)	sa-east-1

Concepts associés à 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 réalisée par le service Amazon ECS exécute un conteneur de 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 Amazon ECS configurées dans d'autres espaces de noms
- Tâches Amazon ECS qui ne sont pas 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. Pour obtenir la liste des méthodes d'interconnexion, reportez-vous à la section Choix d'une méthode d'interconnexion.

Terminologie de Service Connect

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 entre différents clusters Amazon ECS au sein de la même Région AWS dans le même Compte AWS. Étant donné que chaque cluster peut exécuter des tâches de tous les systèmes

d'exploitation, de toutes les architectures de processeur, de tous les VPC et de tous les types de lancement EC2, Fargate et externes, vous pouvez organiser librement vos services selon les critères de votre choix.

service client

Service Amazon ECS qui exécute une application client 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 configurée. Un backend, un intergiciel, une couche entreprise ou la plupart des microservices utilisent 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.

Configuration de cluster

Vous pouvez définir un espace de noms par défaut pour Service Connect lorsque vous créez le cluster ou lorsque vous le mettez à jour. Si vous spécifiez un nom d'espace de noms qui n'existe pas dans le même Région AWS 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 de service Service Connect

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.

L'exemple suivant montre chaque type de configuration Service Connect utilisé ensemble dans le même service Amazon ECS. Des commentaires du shell sont fournis, mais notez que la configuration JSON utilisée pour les services Amazon ECS ne prend pas en charge les commentaires.

```
{
    ...
    serviceConnectConfiguration: {
        enabled: true,
        namespace: "internal",
        #config for client services can end here, only these two parameters are
 required.
        services: [{
                portName: "http"
            }, #minimal client - server service config can end here.portName must match
 the "name"
            parameter of a port mapping in the task definition. {
                discoveryName: "http-second"
                #name the discoveryName to avoid a Task def port name collision with
 the minimal config in the same Cloud Map namespace
                portName: "http"
            },
            {
                clientAliases: [{
                        dnsName: "db",
                        port: 81
```

```
                    }] #use when the port in Task def is not the port that client apps
 use.Client apps can use http: //db:81 to connect
                    discoveryName: "http-three"
                portName: "http"
            },
            {
                clientAliases: [{
                    dnsName: "db.app",
                    port: 81
                }] #use when the port in Task def is not the port that client apps
 use.duplicates are fine as long as the discoveryName is different.
                discoveryName: "http-four"
                portName: "http",
                ingressPortOverride: 99 #If App should also accept traffic directly on
 Task def port.
            }
        ]
    }
}
```

Ordre de déploiement

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 cliente qui fait 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. Après avoir modifié et mis à jour un service Amazon ECS pour ajouter une configuration Service Connect, 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 avec sa montée en puissance, les nouvelles connexions des clients seront immédiatement équilibrées entre toutes les tâches. Si un service client-serveur est mis à jour, les nouvelles connexions des clients seront immédiatement 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 Amazon ECS dont une configuration Service Connect se trouve 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. Par exemple, un service Amazon ECS qui exécute une application cliente doit être redéployé pour se connecter à un nouveau point de terminaison de serveur de base de données. Démarrez le déploiement client une fois le déploiement terminé pour le serveur.

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 de l'espace de noms peut changer chaque fois qu'un service Amazon ECS est déployé dans l'espace de noms, mais les tâches existantes et les tâches de remplacement continuent de se comporter comme elles le faisaient après le déploiement le plus récent.

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, c'est-à-dire toutes les détections d'annulation du déploiement ou AWS Management Console les AWS SDK et autres méthodes permettant de revenir en arrière et de rétablir le déploiement et la configuration précédents du service principal. AWS CLI 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 les AWS SDK et d'autres méthodes pour revenir en arrière et rétablir

le service principal à son déploiement et à sa configuration précédents. AWS CLI 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

Dans la configuration par défaut, le proxy Service Connect écoute sur le `containerPort` depuis le mappage de ports dans la définition de la tâche. Vous avez besoin de règles dans votre groupe de sécurité pour autoriser le trafic entrant (entrant) vers ce port en provenance des CIDR VPC, ou plus particulièrement 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 `ingressPort0verride` 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 dans le groupe de sécurité Amazon VPC 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 `ingressPort0verride` que vous utilisez.

Vous pouvez envoyer du trafic entre des VPC avec Service Connect. Vous devez prendre en compte les mêmes exigences en matière de règles de table de routage, d'ACL réseau et de groupes de sécurité que celles qui s'appliquent aux deux VPC.

Par exemple, deux clusters créent des tâches dans différents VPC. 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 les ACL du réseau VPC n'autorisent le trafic sur les numéros de port et de port que vous utilisez. `containerPort` `ingressPort0verride`

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 instances EC2 du cluster Service Connect.

Proxy Service Connect

Si vous créez ou mettez à jour un service Amazon ECS 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 de ce conteneur dans le service Amazon ECS. De ce fait, vous pouvez réutiliser les mêmes définitions de tâches entre plusieurs services et espaces de noms Amazon ECS, et vous pouvez également exécuter des tâches sans Service Connect.

Ressource de proxy

- Les limites de processeur et de mémoire de la tâche sont les seuls paramètres que vous devez configurer pour ce conteneur dans la définition de tâche. Le seul paramètre que vous devez configurer pour ce conteneur dans le service Amazon ECS est la configuration du journal, que vous trouverez dans la configuration de Service Connect. Pour plus d'informations sur la configuration de Service Connect, veuillez consulter Configuration de service Service Connect.
- La définition de la tâche doit définir la limite de mémoire des tâches pour utiliser Service Connect. Le processeur et la mémoire supplémentaires inclus dans les limites de tâches que vous n'allouez pas dans les limites de conteneur de vos autres conteneurs sont utilisés par le conteneur de proxy Service Connect et les autres conteneurs qui ne définissent pas de limites de conteneur.
- Nous vous recommandons d'ajouter 256 unités de processeur et au moins 64 Mio de mémoire à votre processeur de tâche et à la mémoire pour le conteneur de proxy Service Connect. Sur AWS Fargate, la quantité de mémoire minimale que vous pouvez définir est de 512 Mo de mémoire. Sur Amazon EC2, la mémoire des tâches est facultative, mais elle est requise pour 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 ne se connectent au proxy que si l'application se connecte 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 de service AWS 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 Amazon ECS 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. En raison de la configuration de Service Connect, les connexions à ce nom sont dirigées vers le proxy qui s'exécute dans un conteneur sidecar 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.

Prenons l'exemple d'une tâche exécutée WordPress dans un service Amazon ECS 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 le cadre de WordPress cette tâche, l' WordPress application se connecte au proxy qui

s'exécute dans un conteneur annexe au cours de la même tâche. Le proxy peut se connecter à l'une ou l'autre 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 restera 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. Une valeur de `0` désactive. `perRequestTimeout` 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`.

Remarques relatives à Service Connect

- Les conteneurs Windows ne sont pas pris en charge par Service Connect.

- Les tâches qui s'exécutent dans Fargate doivent utiliser la version 1.4.0 ou une version ultérieure de la plateforme Linux Fargate pour utiliser Service Connect.
- La version de l'agent ECS sur l'instance de conteneur doit être la version 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 plus d’informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.
- L'instance de conteneur `External` pour Amazon ECS Anywhere n'est pas prise en charge par Service Connect.
- Seuls les services qui utilisent des déploiements continus sont pris en charge par Service Connect. Les services qui utilisent les types de déploiement bleu/vert et externe ne sont pas pris en charge.
- Les définitions de tâches doivent définir la limite de mémoire des tâches pour utiliser Service Connect. Pour plus d’informations, consultez Proxy Service Connect.
- Les définitions de tâches qui définissent les limites de mémoire des conteneurs pour tous les conteneurs au lieu de définir la limite de mémoire des tâches 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 plus d’informations, consultez Proxy Service Connect.
- Vous pouvez configurer Service Connect dans un service pour utiliser n'importe quel AWS Cloud Map espace de noms du même Région AWS service. Compte AWS
- Chaque service Amazon ECS ne peut appartenir qu'à un seul espace de noms.
- Seules les tâches créées par les services Amazon ECS sont prises en charge. Les tâches autonomes ne peuvent pas être configurées pour Service Connect.
- 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.
- Les services existants doivent être redéployés avant que les applications qu'ils contiennent 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 plus d'informations, consultez the section called “Ordre de déploiement”.
- Vous pouvez créer un espace de noms lors de la création d'un cluster. Amazon ECS Service Connect ne supprime pas les espaces de noms lorsque des clusters sont supprimés. Vous devez supprimer les espaces de noms directement AWS Cloud Map si vous avez fini de les utiliser.
- Service Connect ne prend pas en charge HTTP 1.0.
- 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.

Expérience de la console Service Connect

Pour créer un nouvel espace de noms, créez un nouveau cluster Amazon ECS à l'aide de la console Amazon ECS et spécifiez un nom d'espace de noms à créer, ou utilisez la AWS Cloud Map console. Amazon ECS Service Connect peut utiliser n'importe quel type d'espace de AWS Cloud Map noms de découverte d'instance. Nous recommandons le type d'appels d'API pour générer le minimum de ressources supplémentaires. Pour créer un nouvel un espace de noms et un nouveau cluster Amazon ECS dans la console Amazon ECS, consultez Création d'un cluster pour le type de lancement Fargate et External à l'aide de la console.

Chaque espace de AWS Cloud Map noms figurant Compte AWS dans le champ sélectionné Région AWS est affiché dans les espaces de noms de la console Amazon ECS.

Pour supprimer un espace de noms, utilisez la AWS Cloud Map console. Un espace de noms doit être vide avant de pouvoir être supprimé.

Pour créer une nouvelle définition de tâche Amazon ECS ou enregistrer une nouvelle révision d'une définition de tâche existante et utiliser Service Connect, consultez Création d'une définition de tâche à l'aide de la console.

Pour créer un service Amazon ECS qui utilise Service Connect, consultez Création d'un service à l'aide de la console.

Tarification de Proxy Service

La tarification d'Amazon ECS Service Connect varie selon que vous utilisez AWS Fargate ou non l'infrastructure Amazon EC2 pour héberger vos charges de travail conteneurisées. Lorsque vous utilisez Amazon ECS sur Amazon EC2 AWS Outposts, la tarification suit le même modèle que celui utilisé lorsque vous utilisez directement Amazon EC2. Pour plus d'informations, consultez Tarification Amazon ECS.

AWS Cloud Map son utilisation est totalement gratuite lorsqu'elle est consommée via Amazon ECS Service Connect.

TLS avec 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 Amazon ECS 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. Par défaut, le protocole TLS 1.3 est pris en charge, mais pas le protocole TLS 1.0 à 1.2. Vous avez également un contrôle total sur les certificats privés et les clés de signature pour vous aider à respecter les exigences de conformité.

> **Note**
>
> Seul le trafic entrant et sortant passant par l'agent Amazon ECS Service Connect est chiffré.

AWS Private Certificate Authority certificats et Service Connect

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 durée de 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`.

Autorités de certification subordonnées

Vous pouvez utiliser n'importe quel protocole TLS AWS Private CA, root ou subordonné, sur Amazon ECS Service Connect pour 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 délivrer des certificats d'entité finale pour différentes parties de votre application à partir de différentes autorités de certification subordonnées. 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**
>
> Veuillez ajouter le tag requis `AmazonECSManaged : true` dans votre AWS Private CA.

Politique IAM d'AmazonECS InfrastructureRolePolicyForServiceConnectTransport LayerSecurity

Amazon ECS fournit une nouvelle politique de confiance en matière de ressources gérées qui décrit l'ensemble des autorisations requises pour l'émission de certificats. Pour plus d'informations sur cette nouvelle politique, consultez AmazonECS InfrastructureRolePolicyForServiceConnectTransport LayerSecurity dans les politiques AWS gérées pour Amazon ECS.

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 valeurs *rouges* par les vôtres.

```
{
  "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 Identification des clés symétriques et AWS 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.

Activer le protocole TLS avec 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. Choisissez l'espace de noms avec le service pour lequel vous souhaitez activer le chiffrement du trafic.
2. Choisissez le service pour lequel vous souhaitez activer le chiffrement du trafic.
3. Choisissez Update Service dans le coin supérieur droit et faites défiler la page vers le bas jusqu'à la section Service Connect.
4. Choisissez Activer le chiffrement du trafic sous les informations de service pour activer le protocole TLS.
5. Pour le rôle TLS Service Connect, choisissez un rôle existant ou créez-en un nouveau.

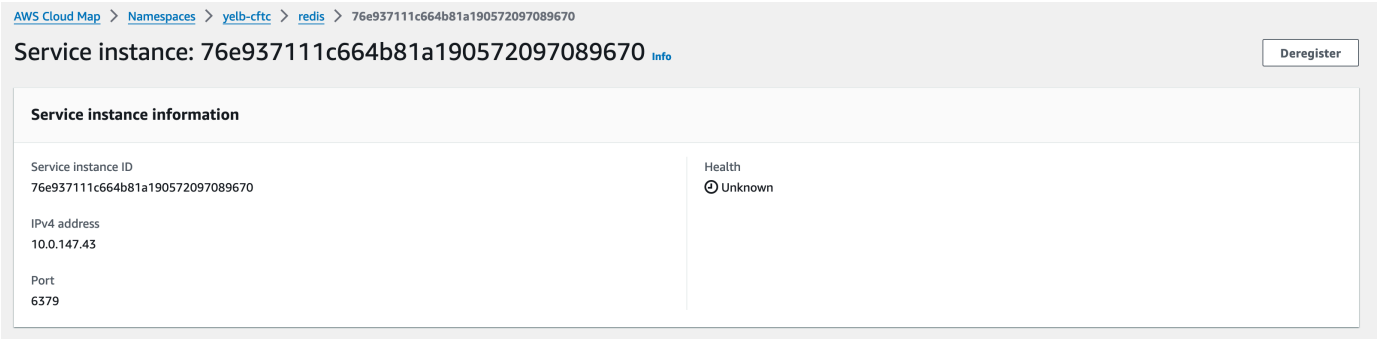
6. Pour Autorité de certification Signer, choisissez une autorité de certification existante ou créez-en une nouvelle.
7. Pour Choisir une AWS KMS key, choisissez une clé AWS détenue et gérée ou vous pouvez choisir une autre clé. Vous pouvez également choisir d'en créer un nouveau.

Pour un exemple d'utilisation du AWS CLI pour configurer le TLS pour votre service, voir Tutoriel : Utilisation de Service Connect dans Fargate avec le. AWS CLI

Vérifiez que le protocole TLS est activé

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. Pour vérifier que le protocole TLS est activé, vous pouvez suivre les étapes suivantes.

1. Assurez-vous que l'image de votre application possède la `openssl` CLI.
2. Activez ECS Exec sur vos services pour vous connecter à vos tâches via SSM. Vous pouvez également créer une instance Amazon EC2 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. Par exemple, si le protocole TLS est activé sur votre `redis` service, vous pouvez récupérer l'adresse IP de sa tâche en accédant au service AWS Cloud Map, en le trouvant et en examinant l'adresse IP et le port d'une instance.



4. Connectez-vous à l'une de vos tâches Amazon ECS `execute-command` comme dans l'exemple suivant. Vous pouvez également vous connecter à l'instance Amazon EC2 créée à l'étape 2.

```
$ aws ecs execute-command --cluster cluster-name \
    --task < TASK_ID>  \
    --container app \
    --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>
```

Paramètres de Service Connect

Les paramètres suivants comportent des champs supplémentaires lors de l'utilisation de Service Connect.

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 plus d'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	Oui

Emplacement des paramètres	Type d'application	Description	Obligatoire ?
		contenir tous les services de serveur que ce service doit découvrir. Pour plus d'informations, consultez `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 plus d'informations, consultez `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. Pour plus d'informations, veuillez consulter Clusters Amazon ECS.	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

Utilisation de Service Connect à Fargate avec le AWS CLI

Le didacticiel suivant explique comment créer un service Amazon ECS avec une tâche Fargate qui utilise Service Connect avec le. AWS CLI

Amazon ECS prend en charge la fonctionnalité Service Connect Régions AWS répertoriée dansRegions with Service Connect.

Prérequis

Le didacticiel suppose de remplir les prérequis suivants :

- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations, consultez Installing the AWS Command Line Interface (Installation de).
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre AWS utilisateur 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 plus d’informations, consultez 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 plus d’informations, consultez Création du rôle `ecsTaskExecutionRole` d'exécution de tâche.

Étape 1 : Créer le cluster Amazon ECS

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 --region us-west-2 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 Amazon ECS 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.
- La définition du service Amazon ECS qui configure Service Connect pour fournir la découverte de services et la mise en proxy de maillage de services 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 Amazon ECS 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 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 ECS en procédant comme suit :

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 les ID du groupe de sécurité et du sous-réseau à partir de 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 ECS à l'aide du fichier `service-connect-nginx-service.json` :

```
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 métriques supplémentaires CloudWatch créées par le proxy Service Connect.

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 instance Amazon EC2 dans le 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, HTTP/2, 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 ECS. Pour plus d'informations sur ces métriques et les dimensions, veuillez consulter Mesures et dimensions disponibles pour Amazon ECS.

Découverte de service

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, voir Qu'est-ce que c'est AWS Cloud Map ? dans le Guide AWS Cloud Map du développeur.

La découverte de services est disponible dans les AWS régions suivantes :

Nom de la région	Région
US East (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
Asie-Pacifique (Sydney)	ap-southeast-2
Asie-Pacifique (Jakarta)	ap-southeast-3
Asie-Pacifique (Melbourne)	ap-southeast-4
Canada (Centre)	ca-central-1

Nom de la région	Région
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
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 la découverte de service : groupe logique de services de découverte de service qui partagent le même nom de domaine, par exemple, `example.com`. Il s'agit du nom de domaine vers lequel vous souhaitez acheminer le trafic. 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'adresse IPv4 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 qui utilisent le type de lancement EC2, il s'agit de la zone de disponibilité dans laquelle l'instance de conteneur existe. 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 plus d'informations, consultez Versions de plateforme Linux Fargate.
- 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.

Protégez vos tâches Amazon ECS pour éviter qu'elles ne soient interrompues par des événements évolutifs

Vous pouvez utiliser la protection évolutive des tâches Amazon ECS pour protéger vos tâches contre l'interruption d'événements de mise à l'échelle horizontale provenant de Service Auto Scaling ou de 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`. Par défaut, les tâches sont protégées pendant 2 heures. Vous pouvez personnaliser la période de protection à l'aide de l'attribut

`expiresInMinutes`. Vous pouvez protéger vos tâches pendant au moins une minute et jusqu'à un maximum de 2 880 minutes (48 heures).

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.

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 :

- 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 instances Amazon EC2 utilisant des versions plus anciennes de l'agent de conteneur Amazon ECS en mettant à jour l'agent à la dernière version. Pour plus d’informations, consultez 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 mise à jour bleu/vert est appliquée, le déploiement bleu contenant les tâches protégées ne sera pas supprimé si les tâches ont `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 demande 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}'
```

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
```

```
{
    "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"
  }
}
```

Logique de régulation du service Amazon ECS

Le planificateur de service Amazon ECS service inclut désormais une logique qui limite la fréquence de lancement des tâches de service en cas d'échec de lancement répété.

Si les tâches d'un service échouent à plusieurs reprises à passer à l'`RUNNING`état (passant directement du `STOPPED` statut a `PENDING` au statut), le délai entre les tentatives de redémarrage suivantes est augmenté progressivement jusqu'à un maximum de 27 minutes. Cette période maximale est sujette à changement à l'avenir. Ce comportement permet de réduire l'effet des tâches défaillantes sur vos ressources de cluster Amazon ECS ou les coûts d'infrastructure Fargate. Si votre service déclenche la logique de limitation, vous recevez le message d'événement de service suivant :

```
(service service-name) is unable to consistently start tasks successfully.
```

Amazon ECS n'empêche jamais un service défaillant de faire une nouvelle tentative. Il ne tente pas non plus de le modifier autrement qu'en augmentant le temps entre les redémarrages. La logique de limitation de service ne fournit pas de paramètres modifiables par l'utilisateur.

Si vous mettez à jour le service de façon à utiliser une nouvelle définition de tâche, celui-ci renvoie un état normal et non limité immédiatement. Pour plus d'informations, consultez Mise à jour d'un service à l'aide de la console.

Les causes les plus courantes à l'origine de cette logique sont les suivantes. Nous vous recommandons de prendre des mesures manuelles pour résoudre le problème :

- Un manque de ressources pour héberger votre tâche, par exemple les ports, la mémoire ou les UC dans votre cluster. Dans ce cas, le message d'événement de service pour ressource insuffisante peut également s'afficher.
- L'agent de conteneur Amazon ECS ne peut pas récupérer l'image Docker de la tâche. Cela peut être dû à un nom de l'image de conteneur, à une image, ou à une étiquette erronés, ou à un manque d'authentification ou d'autorisations de registre privé. Dans ce cas, vous pouvez également voir `CannotPullContainerError` dans les erreurs de tâche interrompue.
- Un espace disque insuffisant sur l'instance de conteneur pour créer le conteneur. Dans ce cas, vous pouvez également voir `CannotCreateContainerError` dans les erreurs de tâche interrompue. Pour plus d'informations, consultez `CannotCreateContainerError: API error (500): devmapper`.

> **Important**
>
> Les tâches qui sont arrêtées après avoir atteint l'état `RUNNING` ne déclenchent pas la limitation logique ou le message d'événement de service associé. Par exemple, supposons que l'échec des surveillances de l'état Elastic Load Balancing pour un service entraîne le

signalement d'une tâche comme défectueuse et qu'Amazon ECS annule son enregistrement et arrête la tâche. À ce stade, les tâches ne sont pas limitées. Même si la commande du conteneur d'une tâche se termine immédiatement avec un code de sortie autre que zéro, la tâche est déjà passée à l'état `RUNNING`. Les tâches qui échouent immédiatement en raison d'erreurs de commande ne déclenchent pas de limitation ou de message d'événement de service.

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 plus d'informations, consultez the section called “Rapports d'utilisation”.

> **Warning**
>
> De nombreuses API renvoient les clés de balise et leurs valeurs. Le fait de refuser l’accès à `DescribeTags` ne refuse pas automatiquement l’accès aux balises renvoyées par d’autres API. 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 plus d’informations, consultez 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.

Prise en charge du balisage pour les ressources Amazon ECS

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, à partir de l'instance Amazon EC2. Pour plus d'informations, consultez Ajout de balises à une instance de conteneur Amazon EC2.	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 plus d’informations, consultez 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 associé 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 plus d'informations, consultez Amazon Resource Names (ARN) et ID.
- Lorsque vous utilisez les API pour créer un service ou exécuter une tâche, vous devez définir `enableECSManagedTags` sur `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.

Identification de vos ressources pour 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 plus d'informations, consultez 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 plus d'informations, consultez 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.

Gestion des étiquettes à l'aide de la console

Vous pouvez utiliser la console Amazon ECS, vous pouvez gérer les balises associées à des tâches, des services, des définitions de tâches, des clusters ou des instances de conteneur nouveaux ou existants.

Lorsque vous sélectionnez une page spécifique d'une ressource sur la console Amazon ECS, celle-ci affiche une liste de ces ressources. Par exemple, si vous sélectionnez Clusters dans le panneau de navigation, la console affiche une liste de clusters Amazon ECS. Lorsque vous sélectionnez une ressource de l'une de ces listes (par exemple, un cluster spécifique) qui prend en charge les balises, vous pouvez afficher et gérer ses balises dans l'onglet Tags (Balises).

> **Warning**
>
> Nous vous recommandons de ne pas inclure de données sensibles dans vos balises.

Table des matières

- Ajout de balises sur une ressource individuelle lors du lancement
- Gestion des balises de ressources individuelles

Ajout de balises sur une ressource individuelle lors du lancement

Vous pouvez utiliser les ressources suivantes pour spécifier des balises lorsque vous créez la ressource.

Tâche	Console

Tâche	Console
Exécuter une ou plusieurs tâches.	Exécuter une application en tant que tâche Amazon ECS
Créer un service.	Création d'un service à l'aide de la console
Créer un ensemble de tâches.	Utiliser un contrôleur tiers pour les déploiements externes d'Amazon ECS
Enregistrer une définition de tâche.	the section called “Création d'une définition de tâche à l'aide de la console”
Créer un cluster.	Création d'un cluster pour le type de lancement Fargate et External à l'aide de la console
Exécuter une ou plusieurs instances de conteneur.	Lancement d'une instance de conteneur Amazon ECS Linux

Gestion des balises de ressources individuelles

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 plus d'informations sur l'identification des instances de conteneur, consultez Ajout de balises à une instance de conteneur Amazon EC2.

> **Warning**
>
> N'ajoutez pas de données d'identification personnelle (PII) ou d'autres informations confidentielles ou sensibles dans les étiquettes. Les étiquettes accessibles à de nombreux

services AWS , y compris la facturation. Les étiquettes ne sont pas destinées à être utilisées pour des données privées ou sensibles.

Pour modifier une balise à une ressource individuelle

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 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.
6. Choisissez Enregistrer.

Ajout de balises à une instance de conteneur Amazon EC2

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 avec l'API, la CLI ou la console Amazon EC2, 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 qui sont associées uniquement à l'instance de conteneur dans Amazon ECS ; elles ne peuvent pas être répertoriés à l'aide de l'API Amazon EC2. Pour plus d'informations, consultez Démarrage des instances de conteneur avec les données utilisateur d'Amazon EC2.

> **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 instances Amazon EC2 lancées à l'aide des 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'API Amazon EC2, de la CLI ou de la console, spécifiez d'abord les balises à l'aide du paramètre `TagSpecification.N`. 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 depuis Amazon EC2 vers Amazon ECS.

  L'exemple suivant présente un script de données utilisateur qui propage les balises associées à une instance Amazon EC2 et enregistre l'instance avec 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 donner l'autorisation aux balises de l'instance de conteneur de propager les données depuis Amazon EC2 vers Amazon ECS, ajoutez manuellement les autorisations suivantes comme stratégie en ligne 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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
             "ec2:DescribeTags"
            ],
            "Resource": "*"
        }
    ]
}
```

Marquage d'une instance de conteneur externe

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 plus d'informations, consultez Gestion des balises de ressources individuelles.

Gestion des balises à l'aide de la CLI ou de l'API

Vous pouvez utiliser l'API AWS Command Line Interface or pour baliser vos ressources Amazon ECS.

Utilisez les commandes suivantes pour ajouter, mettre à jour, répertorier et supprimer les étiquettes pour vos ressources. La documentation correspondante fournit des exemples.

> **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 nombreuses personnes Services AWS, y compris pour la facturation. Les étiquettes ne sont pas destinées à être utilisées pour des données privées ou sensibles.

Prise en charge du balisage pour les ressources Amazon ECS

Tâche	AWS CLI	Action d'API
Ajouter ou remplacer une ou plusieurs étiquettes.	tag-resource	TagResource
Supprimer une ou plusieurs étiquettes.	untag-resource	UntagResource

Vous pouvez utiliser certaines actions de création de ressources pour spécifier des étiquettes lorsque vous créez la ressource. Les actions suivantes prennent en charge l'identification lors de la création.

Vous devez avoir l'autorisation `ecsTagResource`. Pour plus d'informations, consultez Octroi de l'autorisation de baliser les ressources lors de la création.

Tâche	AWS CLI	AWS Tools for Windows PowerShell	Action d'API
Exécuter une ou plusieurs tâches.	run-task	Start-ECSTask	RunTask
Créer un service.	create-service	New-ECSService	CreateService

Tâche	AWS CLI	AWS Tools for Windows PowerShell	Action d'API
Créer un ensemble de tâches.	create-task-set	Nouveaux ECS TaskSet	CreateTaskSet
Enregistrer une définition de tâche.	register-task-definition	Registrer-ECS TaskDefinition	RegisterTaskDefinition
Créer un cluster.	create-cluster	New-ECSCluster	CreateCluster
Exécuter une ou plusieurs instances de conteneur.	run-instances	New-EC2Instance	RunInstances

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 utilisant le type de lancement EC2, la colonne `lineItem/Operation` affiche `ECSTask-EC2` et les tâches ne seront pas associées à un coût direct. 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'instances Amazon EC2. Les données de coût et d'utilisation de vos instances Amazon EC2 seront répertoriées séparément sous le service Amazon EC2.

Vous pouvez également utiliser les balises gérées par Amazon ECS pour identifier le service ou cluster auquel chaque tâche appartient. Pour plus d'informations, consultez Identification de vos ressources pour 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 des articles divisés dans le guide de AWS Cost and Usage Report l'utilisateur

Pour les tâches sur EC2, ces données fractionnent le coût de l'instance EC2 en fonction de l'utilisation des ressources ou des réservations et des ressources restantes de l'instance.

Prérequis

- 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.
- 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.
- Assurez-vous que la `ECS_DISABLE_METRICS` configuration de l'agent Amazon ECS est définie 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.

> **Note**
>
> 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.

Configuration des rapports d'utilisation et de coût au niveau des tâches

Vous pouvez activer les données de répartition des coûts pour ECS dans la console de gestion des coûts ou dans les AWS SDK. AWS Command Line Interface

Utilisez ce qui suit pour les données de répartition des coûts fractionnés.

1. Choisissez de fractionner les données de répartition des coûts. Pour plus d'informations, consultez la section Activation des données de répartition des coûts fractionnés dans le Guide de AWS Cost and Usage Report l'utilisateur.
2. Incluez les données dans un rapport nouveau ou existant.
3. Consultez le rapport. Vous pouvez utiliser la console de Facturation et gestion des coûts ou consulter les fichiers de rapports dans Amazon Simple Storage Service.

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 lancement Amazon EC2, vous devez surveiller les instances EC2 que vous possédez et 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 plus d’informations, consultez Affichage des métriques Amazon ECS.

Rubriques

- Bonnes pratiques pour la surveillance d'Amazon ECS
- Outils de surveillance pour Amazon ECS
- Surveillez Amazon ECS à l'aide de CloudWatch
- Automatisez les réponses aux erreurs Amazon ECS à l'aide de EventBridge
- Surveillez les conteneurs Amazon ECS à l'aide de Container Insights
- Déterminer l'état des tâches Amazon ECS à l'aide de vérifications de l'état des conteneurs
- Surveiller l'état de l'instance de conteneur Amazon ECS
- Identifiez les opportunités d'optimisation d'Amazon ECS à l'aide des données de suivi des applications
- Corrélez les performances des applications Amazon ECS à l'aide des métriques des applications
- Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail
- Identifiez les comportements non autorisés grâce à la surveillance du temps d'exécution
- Surveillez les conteneurs Amazon ECS avec ECS Exec
- AWS Compute Optimizer recommandations pour 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 deviennent graves
- 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 plus d'informations, consultez Emplacements des fichiers journaux Amazon ECS.

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 plus d’informations, consultez 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 plus d’informations, consultez Faites évoluer automatiquement votre service Amazon ECS.

  Pour les clusters dont les tâches ou les services utilisent le type de lancement EC2, 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 plus d’informations, consultez Autorisations requises pour surveiller les instances de conteneur.

- 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 plus d'informations, consultez Utilisation du pilote du journal awslogs.

  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 employant 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 Qu'est-ce qu'Amazon CloudWatch Events ? dans le guide de l'utilisateur d'Amazon CloudWatch Events.
- 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 de cluster pour le type de lancement EC2
  - 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 utiliser CloudWatch pour 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 plus d'informations, consultez 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 plus d’informations, consultez 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 à ce sujet 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.

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 instances Amazon EC2, l'instance Amazon EC2 nécessite la `1.4.0` version ou ultérieure (Linux) `1.0.0` ou ultérieure (Windows) de l'agent de conteneur CloudWatch pour que 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 instances Amazon EC2 nécessitent également l'`ecs:StartTelemetrySession`autorisation du rôle IAM avec lequel vous lancez vos instances Amazon EC2. 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 instances Amazon EC2 `ECS_DISABLE_METRICS=true` en définissant la configuration de votre agent de conteneur Amazon ECS. Pour plus d’informations, consultez Configuration de l'agent de conteneur Amazon ECS.

Mesures et dimensions disponibles pour Amazon ECS

Amazon ECS fournit des CloudWatch métriques gratuites que vous pouvez utiliser pour surveiller vos ressources.

Une métrique représente un ensemble chronologique de points de données publiés sur CloudWatch. L'infrastructure sur laquelle vos tâches Amazon ECS sont hébergées dans vos clusters détermine les métriques disponibles.

CloudWatch les métriques ont des espaces de noms, des dimensions et des statistiques. Les espaces de noms sont utilisés pour isoler les métriques. L'espace de noms Amazon ECS est`AWS/ECS`.

Dimensions qui sont des paires nom/valeur faisant partie de l'identité métrique. Les dimensions sont comme des catégories pour les métriques. Les dimensions d'Amazon ECS incluent`ClusterName`, et`ServiceName`.

Les statistiques sont des agrégations de données métriques sur des périodes spécifiques. CloudWatch fournit des statistiques basées sur les points de données métriques fournis par vos données personnalisées ou fournis par d'autres services AWS connexes CloudWatch. Les regroupements sont effectués en utilisant l’espace de noms, le nom métrique, les dimensions et l’unité de mesure des points de données, pendant la période spécifiée.

Amazon ECS envoie les métriques suivantes CloudWatch toutes les minutes. Quand Amazon ECS collecte des métriques, cela concerne plusieurs points de données toutes les minutes. Il les regroupe ensuite au niveau d'un point de données avant d'envoyer les données à CloudWatch. Ainsi CloudWatch, un dénombrement d'échantillons est en fait l'agrégation de plusieurs points de données pendant une minute.

Statistique	Description
`Average`	La valeur de `Sum` / `SampleCount` pendant la période spécifiée. En comparant cette statistique à `Minimum` et à `Maximum`, vous pouvez déterminer l'ampleur d'une métrique et si l'utilisation moyenne est proche de `Minimum` ou de `Maximum`. Cette comparaison vous permet de savoir quand augmenter ou diminuer vos ressources en fonction des besoins.
`Maximum`	La valeur la plus haute observée pendant la période spécifiée. Vous pouvez utiliser cette valeur pour déterminer les volumes d'activité élevés pour votre application.
`Minimum`	La valeur la plus basse observée pendant la période spécifiée. Vous pouvez utiliser cette valeur pour déterminer les faibles volumes d'activité pour votre application.
`SampleCount`	Le compte (nombre) des points de données utilisé pour le calcul statistique.
`Sum`	Toutes les valeurs soumises pour la métrique correspondante ajoutées ensemble. Cette statistique peut être utile pour déterminer le volume total d'une métrique.

Rubriques

- Métriques Amazon ECS
- Dimensions pour les métriques Amazon ECS
- Métriques de réservation de clusters
- Utilisation du cluster
- Utilisation des services
- Nombre de tâches RUNNING dans le service

Métriques Amazon ECS

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 indicateurs d'utilisation du processeur, de la mémoire et du système de fichiers EBS pour faciliter la surveillance de vos services. Pour les tâches hébergées sur des instances EC2, 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 instances Amazon EC2 qui constituent votre infrastructure sous-jacente. Pour plus d'informations sur la surveillance de vos instances Amazon EC2, consultez la section Surveillance d'Amazon EC2 dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

L'espace de `AWS/ECS` noms CloudWatch inclut les métriques suivantes.

`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 tâches Amazon ECS sur le cluster, divisé par le nombre total d'unités de processeur pour toutes les instances Amazon EC2 enregistrées dans le cluster. Seules les instances Amazon EC2 présentes `ACTIVE` ou `DRAINING` leur statut auront une incidence sur les mesures de réservation du processeur. La métrique n'est prise en charge que pour les tâches hébergées sur une instance Amazon EC2.

Raison de l'utilisation : vous pouvez utiliser cette alarme pour déterminer à quel moment il convient d'augmenter la capacité ou d'ajouter de la capacité supplémentaire.

Valeur recommandée : nous vous recommandons de définir cette métrique sur 90 %. Nous vous déconseillons d'utiliser cette alarme lorsque vous avez des fournisseurs de capacité EC2 dont le dimensionnement géré est activé ou lorsque vous avez des fournisseurs de capacité Fargate. Dans ces cas, Amazon ECS gère le dimensionnement en votre nom.

Dimensions valides : `ClusterName`.

Statistiques utiles : moyenne, minimale, maximale

Unité : pourcentage.

`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 instances Amazon EC2 enregistrées dans le cluster. Seules les instances Amazon EC2 présentes `ACTIVE` ou `DRAINING` leur statut auront une incidence sur les mesures 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 au service. La métrique de niveau de service est prise en charge pour les tâches hébergées sur les instances Amazon EC2 et Fargate.

Raison de l'utilisation : vous pouvez utiliser cette métrique pour déterminer à quel moment ajouter du processeur à vos instances ou à vos tâches afin d'éviter tout problème de saturation des ressources ou de performances des applications.

Valeur recommandée : même si la métrique du niveau de service peut dépasser 100 % d'utilisation, nous vous recommandons de définir le seuil entre 90 et 95 %. Cela peut aider à éviter tout impact sur les autres services. Nous vous recommandons de mettre à jour vos définitions de tâches afin de refléter l'utilisation réelle afin d'éviter de futurs problèmes avec les services.

Dimensions valides : `ClusterName`, `ServiceName`.

Statistiques utiles : moyenne, minimale, maximale

Unité : pourcentage.

`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 instances Amazon EC2 enregistrées dans le cluster. Cette métrique ne peut être filtrée que par`ClusterName`. Seules les instances Amazon EC2 présentes `ACTIVE` ou `DRAINING` leur statut auront une incidence sur les mesures 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 instance Amazon EC2.

> **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 mémoire totale.

Raison de l'utilisation : vous pouvez utiliser cette métrique pour déterminer à quel moment il convient de redimensionner le cluster. Si vous atteignez le nombre total d'unités de mémoire pour le cluster, celui-ci peut être incapable de lancer de nouvelles tâches.

Valeur recommandée : nous vous recommandons de définir cette métrique sur 90 %. Nous vous déconseillons d'utiliser cette alarme lorsque vous avez des fournisseurs de capacité EC2 dont le dimensionnement géré est activé ou lorsque vous avez des fournisseurs de capacité Fargate. Dans ces cas, Amazon ECS gère le dimensionnement en votre nom.

Dimensions valides : `ClusterName`.

Statistiques utiles : moyenne, minimale, maximale

Unité : pourcentage.

`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 instances Amazon EC2 enregistrées dans le cluster. Seules les instances Amazon EC2 présentes `ACTIVE` ou en `DRAINING` état auront une incidence sur les mesures d'utilisation de la mémoire. La 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 instances Amazon EC2 et Fargate.

Raison de l'utilisation : vous pouvez utiliser cette métrique pour déterminer à quel moment ajouter de la mémoire à vos instances ou à vos tâches afin d'éviter tout problème de saturation des ressources ou de performances des applications.

Valeur recommandée : nous vous recommandons de définir le seuil à 90-95 %. Cela peut aider à éviter tout impact sur les autres services. Nous vous recommandons de mettre à jour vos définitions de tâches afin de refléter l'utilisation réelle afin d'éviter de futurs problèmes avec les services.

Dimensions valides : `ClusterName`, `ServiceName`.

Statistiques utiles : moyenne, minimale, maximale

Unité : pourcentage.

`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'utilisation du système de fichiers EBS au niveau de service n'est disponible que pour les tâches hébergées sur des instances Amazon EC2 (à l'aide de la version de l'agent de conteneur`1.79.0`) et sur Fargate (à l'aide de la `1.4.0` version de plate-forme) auxquelles un volume EBS est attaché.

> **Note**
>
> Pour les tâches hébergées sur Fargate, il y a de l'espace sur le disque uniquement 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`

Dimensions valides : `ClusterName, ServiceName`.

Statistiques utiles : moyenne, minimale, maximale

Unité : pourcentage.

`GPUReservation`

Le pourcentage du total de GPU disponibles qui sont réservés en exécutant des tâches dans le cluster.

La métrique de réservation de GPU au niveau du cluster est mesurée comme le nombre de GPU réservés par les tâches Amazon ECS sur le cluster, divisé par le nombre total de GPU disponibles sur toutes les instances Amazon EC2 avec des GPU enregistrés dans le cluster. Seules les instances Amazon EC2 présentes `ACTIVE` ou `DRAINING` leur statut auront une incidence sur les métriques de réservation du GPU.

Dimensions valides : `ClusterName`.

Statistiques utiles : moyenne, minimale, maximale

Toutes les statistiques : moyenne, minimale, maximale, somme, nombre d'échantillons.

Unité : pourcentage.

`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`.

Statistiques utiles : moyenne, minimum, maximum, somme.

Unité : nombre.

`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 `DiscoveryName` sélectionné.

Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.

Dimensions valides : `DiscoveryName` et `DiscoveryName, ServiceName, ClusterName`.

Statistiques utiles : moyenne, minimum, maximum, somme.

Unité : nombre.

`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.

Dimensions valides : `DiscoveryName` et `DiscoveryName, ServiceName, ClusterName`.

Statistiques utiles : moyenne, minimum, maximum, somme.

Unité : octets.

`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.

Dimensions valides : `DiscoveryName` et `DiscoveryName, ServiceName, ClusterName`.

Statistiques utiles : moyenne, minimum, maximum, somme.

Unité : nombre.

`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.

Dimensions valides : `DiscoveryName` et `DiscoveryName, ServiceName, ClusterName`.

Statistiques utiles : moyenne, minimum, maximum, somme.

Unité : nombre.

`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 le mappage des ports de la définition de la tâche.

Dimensions valides : `TargetDiscoveryName` et `TargetDiscoveryName, ServiceName, ClusterName`.

Statistiques utiles : moyenne, minimum, maximum, somme.

Unité : nombre.

`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 le mappage des ports de la définition de la tâche.

Dimensions valides : `TargetDiscoveryName` et `TargetDiscoveryName, ServiceName, ClusterName`.

Statistiques utiles : moyenne, minimum, maximum, somme.

Unité : nombre.

`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 le mappage des ports de la définition de la tâche.

Dimensions valides : `TargetDiscoveryName` et `TargetDiscoveryName, ServiceName, ClusterName`.

Statistiques utiles : moyenne, minimum, maximum, somme

Unité : nombre.

`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 le mappage des ports de la définition de la tâche.

Raison de l'utilisation : vous pouvez utiliser cette métrique pour détecter un nombre élevé d'erreurs côté serveur pour un service.

Valeur recommandée : nous vous recommandons de définir initialement la valeur à 5 % de votre trafic moyen. Vous pouvez utiliser la `RequestCount` métrique pour trouver le trafic moyen. Vous devrez peut-être ajuster la valeur pour que l'alarme ne soit pas trop sensible.

Statistiques utiles : moyenne, minimum, maximum, somme.

Unité : nombre.

`RequestCountPerTarget`

Nombre moyen de demandes reçues par chaque cible qui partage le `DiscoveryName` sélectionné.

Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.

Dimensions valides : `TargetDiscoveryName` et `TargetDiscoveryName, ServiceName, ClusterName`.

Statistiques utiles : moyenne.

Unité : 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.

Dimensions valides : `TargetDiscoveryName` et `TargetDiscoveryName, ServiceName, ClusterName`.

Statistiques utiles : moyenne, minimum, maximum, somme.

Unité : octets.

`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.

Raison de l'utilisation : vous pouvez utiliser cette métrique pour détecter un temps de réponse élevé du service. Vous pouvez ensuite examiner d'autres indicateurs, par exemple `CPUUtilization` pour déterminer si le service dispose de suffisamment de ressources.

Valeur recommandée : passez en revue le temps de réponse critique et le comportement historique de la métrique pour déterminer une valeur.

Dimensions valides : `TargetDiscoveryName` et `TargetDiscoveryName, ServiceName, ClusterName`.

Statistiques utiles : moyenne, minimale, maximale.

Toutes les statistiques : moyenne, minimale, maximale, somme, nombre d'échantillons.

Unité : millisecondes.

`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.

Dimensions valides : `DiscoveryName` et `DiscoveryNameServiceName`,`ClusterName`.

Statistiques utiles : moyenne, minimum, maximum, somme.

Unité : nombre.

`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.

Dimensions valides :`ServiceName`,`ClusterName`, `TargetDiscoveryName` et`TargetDiscoveryName`.

Statistiques utiles : moyenne, minimum, maximum, somme.

Unité : 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.

`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.

`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.

`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.

Métriques de réservation de clusters

Les métriques de réservation de cluster correspondent au pourcentage d'UC, de mémoire et de GPU réservé par toutes les tâches Amazon ECS sur un cluster par rapport aux ressources d'UC, de mémoire et de GPU regroupées qui ont été enregistrées pour chaque instance de conteneur active dans le 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 des instances EC2. 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)
```

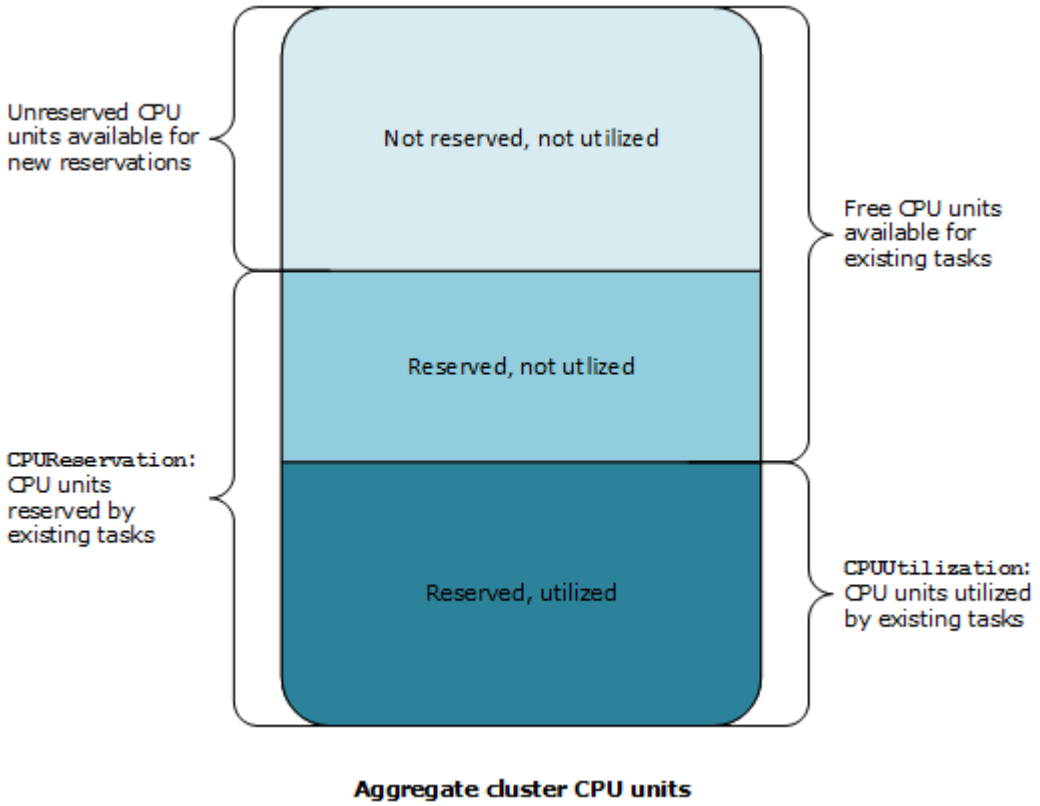
Lorsque vous exécutez une tâche dans un cluster, Amazon ECS analyse la définition de la tâche et réserve les unités d'UC, les Mio de mémoire et les GPU regroupés qui sont spécifiés dans ses définitions de conteneur. Chaque minute, Amazon ECS calcule le nombre d'unités d'UC, les Mio de mémoire et les GPU qui sont actuellement réservés pour chaque tâche en cours d'exécution dans le cluster. La quantité totale de CPU, de mémoire et de GPU réservés 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`.



Utilisation du cluster

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 des instances Amazon EC2. Ils ne sont pas pris en charge sur les clusters sur lesquels les tâches sont hébergées AWS Fargate.

Utilisation du processeur et de la mémoire au niveau du cluster

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 instance Amazon EC2 active enregistrée dans le cluster. Seules les instances Amazon EC2 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 instance Amazon EC2 calcule le nombre d'unités CPU et de MiB de mémoire actuellement utilisés pour chaque tâche exécutée sur cette instance Amazon EC2, 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 instances Amazon EC2 actives enregistrées, une `c4.4xlarge` instance et une `c4.large` instance. L'`c4.4xlarge`instance s'enregistre dans le cluster avec des unités de `16,384` processeur 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 EBS au niveau du cluster

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)
```

Utilisation des services

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 instances Amazon EC2 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.

```
                                      (Total CPU units used by tasks in service) x 100
Service CPU utilization =
 ------------------------------------------------------------------------------
                           (Total CPU units specified in task definition) x (number of
 tasks in service)
```

```
                                        (Total MiB of memory used by tasks in service) x
 100
Service memory utilization =
 ---------------------------------------------------------------------------
                              (Total MiB of memory specified in task definition) x
 (number of tasks in service)
```

Chaque minute, l'agent de conteneur Amazon ECS calcule le nombre d'unités CPU et de MiB de mémoire actuellement utilisés pour chaque tâche appartenant au service, et ces informations sont transmises à Amazon ECS. La quantité totale de CPU et de mémoire utilisée pour toutes les tâches détenues par le service exécutées sur le cluster est calculée, et ces chiffres sont indiqués CloudWatch sous forme de pourcentage des ressources totales spécifiées pour le service dans la définition des tâches du service. 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.

```
                                    (Total GB of EBS filesystem used by tasks in the
 service x 100)
Service EBS filesystem utilization =
 ------------------------------------------------------------------
                                    (Total GB of EBS filesystem allocated to tasks
 in the 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.

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 un quota de service. Pour de plus amples informations sur les Service Quotas pour Fargate, consultez AWS Fargate quotas de service.

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.

Dimension	Description
`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 Service Quotas l'utilisateur

.

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 plus d'informations, consultez Utilisation des services.

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
2. Sélectionnez le cluster dans lequel vous voulez afficher les métriques.
3. Déterminez les indicateurs à consulter.

Pour consulter les métriques 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 lancement EC2, 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 la section Afficher les mesures disponibles dans le guide de CloudWatch l'utilisateur Amazon.

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
- Appel de la fonctionnalité Exécuter la commande d'Amazon EC2
- 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 elles utilisent le type de lancement EC2, vous pouvez voir l'état des instances de conteneur et l'état actuel de toutes les tâches exécutées sur ces instances. 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 plus d'informations, consultez 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 a généré et envoyé les types d'événements suivants à EventBridge : événements de changement d'état d'instance de conteneur, événements de changement d'état de tâche, événements de changement d'état de service et événements de changement d'état de déploiement de services.

- 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 peut ajouter d'autres types d'événements, de sources et de détails par la suite. 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 la section Événements et modèles d'événements 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 les API Amazon ECS avec la version indiquée 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.

Pour plus d'informations sur la manière d'intégrer Amazon ECS EventBridge, consultez Intégrer Amazon EventBridge et 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 d'API `StartTask`, `RunTask` ou `StopTask` directement ou avec la AWS Management Console ou les kits SDK.

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 d'`DeregisterContainerInstance`API, soit directement, soit avec les SDK AWS Management Console .

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 s'est arrêtée en même temps qu'une instance EC2.

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 la section Événements et modèles d'événements 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 d'API `StartTask`, `RunTask` ou `StopTask` directement ou avec AWS Management Console, la AWS CLI ou les kits SDK.

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 lancement Fargate, l'agent de conteneur Amazon ECS surveille l'état de vos instances de conteneur. Pour le type de lancement Amazon EC2, 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 les AWS Management Console SDK.

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. La `detail` section ci-dessous ressemble à l'objet Task renvoyé par une opération d'DescribeTasksAPI dans le Amazon Elastic Container Service API Reference. Si vos conteneurs utilisent une image hébergée avec Amazon ECR, le champ `imageDigest` est renvoyé.

> **Note**
>
> Les valeurs des champs `createdAt`, `connectivityAt`, `pullStartedAt`, `startedAt`, `pullStoppedAt` et `updatedAt` sont des horodatages UNIX dans la réponse d'une

action `DescribeTasks`, mais sont des horodatages de chaîne ISO dans l'événement de changement d'état de tâche.

Pour plus d'informations sur les paramètres CloudWatch des événements, consultez la section Événements et modèles d'événements dans le guide de EventBridge l'utilisateur Amazon.

Pour plus d'informations sur la configuration et la configuration d'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, voir Envoi d'alertes Amazon Simple Notification Service en cas d'arrêt d'une tâche

```
{
    "version": "0",
    "id": "3317b2af-7005-947d-b652-f55e762e571a",
    "detail-type": "ECS Task State Change",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2020-01-23T17:57:58Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:task/FargateCluster/
c13b4cb40f1f4fe4a2971f76ae5a47ad"
    ],
    "detail": {
        "attachments": [
            {
                "id": "1789bcae-ddfb-4d10-8ebe-8ac87ddba5b8",
                "type": "eni",
                "status": "ATTACHED",
                "details": [
                    {
                        "name": "subnetId",
                        "value": "subnet-abcd1234"
                    },
                    {
                        "name": "networkInterfaceId",
                        "value": "eni-abcd1234"
                    },
                    {
                        "name": "macAddress",
                        "value": "0a:98:eb:a7:29:ba"
```

```
                    },
                    {
                        "name": "privateIPv4Address",
                        "value": "10.0.0.139"
                    }
                ]
            }
        ],
        "availabilityZone": "us-west-2c",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/FargateCluster",
        "containers": [
            {
                "containerArn": "arn:aws:ecs:us-west-2:111122223333:container/
cf159fd6-3e3f-4a9e-84f9-66cbe726af01",
                "lastStatus": "RUNNING",
                "name": "FargateApp",
                "image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/hello-
repository:latest",
                "imageDigest":
 "sha256:74b2c688c700ec95a93e478cdb959737c148df3fbf5ea706abe0318726e885e6",
                "runtimeId":
 "ad64cbc71c7fb31c55507ec24c9f77947132b03d48d9961115cf24f3b7307e1e",
                "taskArn": "arn:aws:ecs:us-west-2:111122223333:task/FargateCluster/
c13b4cb40f1f4fe4a2971f76ae5a47ad",
                "networkInterfaces": [
                    {
                        "attachmentId": "1789bcae-ddfb-4d10-8ebe-8ac87ddba5b8",
                        "privateIpv4Address": "10.0.0.139"
                    }
                ],
                "cpu": "0"
            }
        ],
        "createdAt": "2020-01-23T17:57:34.402Z",
        "launchType": "FARGATE",
        "cpu": "256",
        "memory": "512",
        "desiredStatus": "RUNNING",
        "group": "family:sample-fargate",
        "lastStatus": "RUNNING",
        "overrides": {
            "containerOverrides": [
                {
                    "name": "FargateApp"
```

```
                }
            ]
        },
        "connectivity": "CONNECTED",
        "connectivityAt": "2020-01-23T17:57:38.453Z",
        "pullStartedAt": "2020-01-23T17:57:52.103Z",
        "startedAt": "2020-01-23T17:57:58.103Z",
        "pullStoppedAt": "2020-01-23T17:57:55.103Z",
        "updatedAt": "2020-01-23T17:57:58.103Z",
        "taskArn": "arn:aws:ecs:us-west-2:111122223333:task/FargateCluster/
c13b4cb40f1f4fe4a2971f76ae5a47ad",
        "taskDefinitionArn": "arn:aws:ecs:us-west-2:111122223333:task-definition/
sample-fargate:1",
        "version": 4,
        "platformVersion": "1.3.0"
    }
}
```

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 la section Création d'une EventBridge règle 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.

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.

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 plus d’informations, consultez Amazon Resource Names (ARN) et ID. Une autre cause fréquente est l'incapacité d'Amazon ECS à assumer le rôle IAM fourni pour la tâche.

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 Événements et modèles d'événements dans le 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 sur EventBridge les paramètres, consultez la section Événements et modèles d'événements dans le 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, pour le type de lancement EC2, la tâche a tenté de se lancer sur l'instance de conteneur `2dd1b186f39845a584488d2ef155c131` mais le planificateur de service n'a pas pu placer la tâche 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, consultez la section Création d'une EventBridge règle 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`. 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 la section Événements et modèles d'événements 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 plus d'informations, consultez La mise à jour propagée.

```
{
   "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 plus d'informations, consultez La mise à jour propagée.

```
{
   "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 les API Amazon ECS 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 :

- ECS CtrInstanceState — Stocke le dernier état d'une instance de conteneur. L'ID de table correspond à la valeur `containerInstanceArn` de l'instance de conteneur.
- ECS TaskState — Stocke 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

CloudWatch Container Insights collecte, agrège et résume les métriques et les journaux de vos applications conteneurisées et de vos microservices.

Container Insights utilise une version conteneurisée de l' CloudWatch agent pour découvrir tous les conteneurs en cours d'exécution dans un cluster et collecter des données de performance à chaque niveau 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. Pour plus d'informations sur les métriques disponibles, consultez les métriques Amazon ECS Container Insights dans le guide de CloudWatch l'utilisateur Amazon.

> **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. Amazon ECS propose également des métriques de surveillance sans coûts supplémentaires. Pour plus d'informations, consultez Surveillez Amazon ECS à l'aide de CloudWatch .

Considérations

Les points suivants doivent être pris en compte lors de l'utilisation de CloudWatch Container Insights.

- CloudWatch Les métriques de Container Insights 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.

- Les métriques réseau sont disponibles pour toutes les tâches exécutées sur Fargate et les tâches exécutées sur des instances Amazon EC2 qui utilisent les modes réseaux `bridge` ou `awsvpc`.

Vous pouvez consulter les événements du cycle de vie des tâches et des services Amazon ECS dans la console CloudWatch Container Insights. Cela vous aide à corréler vos métriques, journaux et événements de conteneurs en une seule vue pour vous donner une visibilité opérationnelle plus complète.

Les événements que vous pouvez consulter sont ceux qu'Amazon ECS envoie à Amazon EventBridge. Pour plus d'informations, consultez Événements Amazon ECS.

Vous pouvez choisir de configurer des métriques de performance pour les clusters, les tâches ou les services. En fonction de la ressource que vous choisissez, les événements suivants sont signalés :

- Événements de modification de l'état de l'instance de conteneur
- Événements d'action de service
- Événements de modification de l'état de la tâche

Configuration de CloudWatch Container Insights pour Amazon ECS

Vous pouvez configurer Container Insights à l'aide de la console Amazon ECS AWS CLI, de l'API et des SDK.

Utilisez le tableau suivant pour déterminer les mesures à prendre pour ajouter Container Insights.

Prise en charge du balisage pour les ressources Amazon ECS

Tâche	Console	AWS CLI	Action d'API
Modifier la valeur par défaut pour tous les utilisateurs	Modification des paramètres de compte	put-account-setting-default	PutAccountSettingDefault
Modifier la valeur par défaut pour un utilisateur spécifique	Modification des paramètres de compte	put-account-setting	PutAccountSetting

Tâche	Console	AWS CLI	Action d'API
Configurer Container Insights pour un cluster spécifique	Création d'un cluster pour le type de lancement Fargate et External à l'aide de la console Création d'un cluster pour le type de lancement Amazon EC2 à l'aide de la console Mise à jour d'un cluster à l'aide de la console	create-cluster UpdateCluster	CreateCluster UpdateCluster

> **Important**
>
> Pour des clusters contenant des tâches ou des services utilisant le type de lancement EC2, vos instances de conteneur doivent exécuter la version 1.29.0 ou ultérieure de l'agent Amazon ECS. Pour plus d'informations, consultez Gestion des instances de conteneur Linux.

Autorisations requises pour que CloudWatch Container Insights puisse consulter les événements du cycle de vie d'Amazon ECS

Vous devez configurer les autorisations appropriées, puis vous pouvez configurer et afficher les événements dans la console CloudWatch Container Insights. Pour plus d'informations, consultez les événements du cycle de vie d'Amazon ECS dans Container Insights dans le guide de CloudWatch l'utilisateur Amazon. Pour plus d'informations sur les politiques IAM pour CloudWatch, voir AWS Identity and Access Management pour CloudWatch.

Autorisations requises pour configurer Container Insights afin d'afficher les événements du cycle de vie Amazon ECS

Les autorisations suivantes sont requises dans le rôle de tâche pour configurer les événements du cycle de vie :

- événements : PutRule
- événements : PutTargets
- journaux : CreateLogGroup

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutRule",
        "events:PutTargets",
        "logs:CreateLogGroup"
      ],
      "Resource": "*"
    }
  ]
}
```

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 sous forme de stratégie en ligne au rôle d'exécution de tâche. Pour plus d'informations, consultez Ajout et suppression de politiques IAM.

- événements : DescribeRule
- événements : ListTargetsByRule
- journaux : DescribeLogGroups

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:DescribeRule",
        "events:ListTargetsByRule",
        "logs:DescribeLogGroups"
      ],
      "Resource": "*"
    }
  ]
}
```

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.
- 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.

Pour plus d'informations sur la manière de spécifier un bilan de santé dans la définition d'une tâche, consultezSurveillance de l'état.

Les valeurs d'état de santé possibles d'un conteneur sont décrites ci-dessous :

- `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 :

- Lorsque l'agent Amazon ECS ne peut pas se connecter au service Amazon ECS, le service signale le conteneur comme `UNHEALTHY`.
- Les statuts de surveillance de l'état constituent les « dernières nouvelles » de l'agent Amazon ECS. Aucune hypothèse n'a été émise quant au statut des surveillances de l'état 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 estUNHEALTHY, le statut de la tâche estUNHEALTHY.
2. Si le statut d'un conteneur essentiel estUNKNOWN, le statut de la tâche estUNKNOWN.
3. Si le statut de tous les conteneurs essentiels est le HEALTHY même, le statut de la tâche estHEALTHY.

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

Afficher l'état des conteneurs Amazon ECS

Vous pouvez consulter l'état du conteneur dans la console, ainsi que 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, 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.

```
"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.

Les contrôles de statut sont exécutés environ deux fois par minute et chacun d'entre eux renvoie un statut de réussite ou d'échec. Si tous les contrôles réussissent, l'état global de l'instance est `OK`. Si un ou plusieurs contrôles échouent, le statut global de l'instance est `IMPAIRED`. Les contrôles de

statut sont intégrés à l'agent de conteneur Amazon ECS. Ils ne peuvent donc pas être désactivés ou supprimés. Vous pouvez afficher les résultats de ces contrôles de statut pour identifier des problèmes spécifiques et détectables. Pour plus d'informations, consultez the section called "Surveillance de l'état".

Exécutez l'`DescribeContainerInstances`API avec l'`CONTAINER_INSTANCE_HEALTH`option permettant de récupérer l'état de santé de l'instance du conteneur.

```
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"
 }]
}
```

Rubriques en relation

- Surveillez Amazon ECS à l'aide de CloudWatch

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 goulots d'étranglement liés aux performances 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 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. 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 métriques d'application 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 plus d'informations, consultez Corrélez les performances des applications Amazon ECS à l'aide des métriques 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 l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console IAM, choisissez Rôles, puis 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 les limites des autorisations, puis choisissez Utiliser une limite d'autorisations pour contrôler les autorisations maximales des rôles.

      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, tenez compte des points suivants :
   >
   > - 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 de rôles nommés à la fois **`PRODROLE`** et**`prodrole`**. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, il distingue les majuscules et minuscules, mais lorsqu'un nom de rôle apparaît aux clients dans la console, par exemple pendant le processus de connexion, le nom du rôle ne fait pas la distinction entre majuscules et minuscules.
   > - Vous ne pouvez pas modifier le nom du rôle une fois qu'il a été créé car d'autres entités peuvent y faire référence.

10. (Facultatif) Dans Description, entrez une description pour le rôle.

11. (Facultatif) Pour modifier les cas d'utilisation et les autorisations du rôle, dans les sections Étape 1 : Sélection des entités de confiance ou Étape 2 : Ajouter des autorisations, choisissez Modifier.

12. (Facultatif) Pour identifier, organiser ou rechercher le rôle, ajoutez des balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la rubrique Balisage des ressources IAM dans le Guide de l'utilisateur 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 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 plus d'informations, consultez Création d'une définition de tâche 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 for 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 plus d'informations, consultez Création d'une définition de tâche 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 plus d'informations, consultez Surveillez les conteneurs Amazon ECS à l'aide de Container Insights.
- 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 plus d'informations, consultez 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 l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console IAM, choisissez Rôles, puis 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 les limites des autorisations, puis choisissez Utiliser une limite d'autorisations pour contrôler les autorisations maximales des rôles.

      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, tenez compte des points suivants :
   >
   > - 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 de rôles nommés à la fois **`PRODROLE`** et**`prodrole`**. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, il distingue les majuscules et minuscules, mais lorsqu'un nom de rôle apparaît aux clients dans la console, par exemple pendant le processus de connexion, le nom du rôle ne fait pas la distinction entre majuscules et minuscules.
   > - Vous ne pouvez pas modifier le nom du rôle une fois qu'il a été créé car d'autres entités peuvent y faire référence.

10. (Facultatif) Dans Description, entrez une description pour le rôle.

11. (Facultatif) Pour modifier les cas d'utilisation et les autorisations du rôle, dans les sections Étape 1 : Sélection des entités de confiance ou Étape 2 : Ajouter des autorisations, choisissez Modifier.

12. (Facultatif) Pour identifier, organiser ou rechercher le rôle, ajoutez des balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la rubrique Balisage des ressources IAM dans le Guide de l'utilisateur 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 plus d'informations, consultez Création d'une définition de tâche 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 plus d'informations, consultez Création d'une définition de tâche 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 plus d’informations,

consultez 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 l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation de la console IAM, choisissez Rôles, puis 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 les limites des autorisations, puis choisissez Utiliser une limite d'autorisations pour contrôler les autorisations maximales des rôles.

      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, tenez compte des points suivants :
>
> - 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 de rôles nommés à la fois **`PRODROLE`** et**`prodrole`**. Lorsqu'un nom de rôle est utilisé dans une politique ou dans le cadre d'un ARN, il distingue les majuscules et minuscules, mais lorsqu'un nom de rôle apparaît aux

clients dans la console, par exemple pendant le processus de connexion, le nom du rôle ne fait pas la distinction entre majuscules et minuscules.

- Vous ne pouvez pas modifier le nom du rôle une fois qu'il a été créé car d'autres entités peuvent y faire référence.

10. (Facultatif) Dans Description, entrez une description pour le rôle.
11. (Facultatif) Pour modifier les cas d'utilisation et les autorisations du rôle, dans les sections Étape 1 : Sélection des entités de confiance ou Étape 2 : Ajouter des autorisations, choisissez Modifier.
12. (Facultatif) Pour identifier, organiser ou rechercher le rôle, ajoutez des balises sous forme de paires clé-valeur. Pour plus d'informations sur l'utilisation des balises dans IAM, consultez la rubrique Balisage des ressources IAM dans le Guide de l'utilisateur 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 plus d'informations, consultez Création d'une définition de tâche 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 ECS est intégré à AWS CloudTrail un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou un AWS service dans Amazon ECS. CloudTrail capture tous les appels d'API pour Amazon ECS sous forme d'événements, y compris les appels depuis la console Amazon ECS et les appels de code vers les opérations d'API Amazon ECS. Pour protéger votre VPC, les demandes refusées par une politique de point de terminaison du VPC, mais qui auraient autrement été autorisées, ne sont pas enregistrées dans. CloudTrail

Si vous créez un suivi, vous pouvez activer la diffusion continue d' CloudTrail événements vers un compartiment Amazon S3, y compris des événements pour Amazon ECS. Si vous ne configurez pas de suivi, vous pouvez toujours consulter les événements les plus récents dans la CloudTrail console dans Historique des événements. À l'aide des informations collectées par CloudTrail, vous pouvez déterminer la demande qui a été 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 plus d’informations, consultez le Guide de l’utilisateur AWS CloudTrail.

Informations Amazon ECS dans CloudTrail

CloudTrail est activé dans votre AWS compte lorsque vous le créez. Lorsqu'une activité 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 à l'aide de 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. section withinPour plus d'informations, consultez :

- Présentation de la création d’un journal d’activité

- CloudTrail Services et intégrations pris en charge
- Configuration des notifications Amazon SNS pour CloudTrail
- Réception de fichiers CloudTrail journaux de plusieurs régions et réception de fichiers CloudTrail journaux de plusieurs comptes

Toutes les actions Amazon ECS sont enregistrées CloudTrail et documentées dans le manuel Amazon Elastic Container Service API Reference. Par exemple, les appels aux `DeleteCluster` sections`CreateService`, `RunTask` et génèrent des entrées dans les fichiers CloudTrail journaux.

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 effectuée avec les informations d'identification de sécurité temporaires d'un rôle ou d'un utilisateur fédéré.
- Si la demande a été faite par un autre AWS service.

Pour plus d'informations, consultez la section Élément userIdentity CloudTrail .

Présentation des entrées des fichiers journaux Amazon ECS

Un suivi est une configuration qui permet de transmettre des événements sous forme de fichiers journaux à un compartiment Amazon S3 que vous spécifiez. CloudTrail les fichiers journaux contiennent une ou plusieurs entrées de journal. Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la demande, etc. CloudTrail les fichiers journaux ne constituent pas une trace ordonnée des appels d'API publics, ils n'apparaissent donc pas dans un ordre spécifique.

> **Note**
>
> Ces exemples ont été mis en forme pour faciliter la 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.

L'exemple suivant montre une entrée de CloudTrail journal qui illustre l'`CreateCluster`action :


```
{
    "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"
}
```

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 les instances de conteneur Fargate et EC2 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 instances Amazon EC2 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 conteneur EC2

- 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 conteneur EC2.

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 afin de 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 conteneur EC2, 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 TAGS`option.
  - 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 plus d’informations, consultez 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 mémoire d'Amazon EC2, consultez la section Limite de processeur et de mémoire pour GuardDuty l'agent.

GuardDuty gestion des agents

Si vous utilisez des instances de conteneur EC2, vous devez configurer manuellement la surveillance du temps d'exécution. Pour plus d'informations, consultez Gestion manuelle de la surveillance du temps d'exécution.

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.

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.
- 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.
- Rôles et autorisations IAM pour Amazon ECS :
  - Les tâches Fargate doivent utiliser un rôle d'exécution de tâches. Ce rôle accorde aux tâches l'autorisation de récupérer, de mettre à jour et de gérer l'agent GuardDuty de sécurité en votre nom. Pour plus d'informations, consultez 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 la section Utilisation d'un sous-réseau privé et d'une 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`.

Activer la surveillance du temps d'exécution

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 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 protégés automatiquement. Afin de préserver la contrainte d'immuabilité, les tâches existantes ne sont pas affectées. Pour protéger immédiatement une tâche, vous devez effectuer l'une des actions suivantes :

- Pour les tâches autonomes, arrêtez les tâches, puis démarrez-les.
- Pour les tâches faisant partie d'un service, mettez à jour le service avec l'option « forcer un nouveau déploiement ».

Supprimer la surveillance du temps d'exécution d'un cluster

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.

Effectuez les opérations suivantes 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 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.

Gestion manuelle de la surveillance du temps d'exécution

Utilisez-le lorsque vous utilisez des instances EC2 pour votre capacité ou lorsque vous avez besoin d'un contrôle précis de l'activation 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 conteneur EC2, vous téléchargez, installez et gérez l'agent GuardDuty de sécurité.

Pour Fargate GuardDuty , gère l'agent de sécurité en votre nom.

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 accorde aux tâches l'autorisation de récupérer, de mettre à jour et de gérer l'agent GuardDuty de sécurité en votre nom. Pour plus d'informations, consultez 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 la section Utilisation d'un sous-réseau privé et d'une 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 conteneur EC2 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 instances EC2 dans le guide de l'AWS Systems Manager Session Manager utilisateur.

Activer la surveillance du temps d'exécution

Vous activez cette fonctionnalité 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.

Configuration de la surveillance du temps d'exécution pour les clusters

Configurez la surveillance du temps d'exécution pour le cluster, puis installez l'agent GuardDuty de sécurité sur vos instances de conteneur EC2.

Ajouter la surveillance du temps d'exécution à un cluster

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 conteneur EC2.

    a. Mettez à jour la version `1.77` ou ultérieure de l'agent Amazon ECS sur les instances de conteneur EC2 du cluster. Pour plus d’informations, consultez Mise à jour de l'agent de conteneur Amazon ECS.

    b. Installez l'agent GuardDuty de sécurité sur les instances de conteneur EC2 du cluster. Pour plus d'informations, consultez la section Gestion manuelle de l'agent de sécurité sur une instance Amazon EC2 dans le guide de l'GuardDuty utilisateur.

Toutes les tâches et tous les déploiements nouveaux et existants sont immédiatement protégés car l'agent de GuardDuty sécurité s'exécute en tant que processus sur l'instance de conteneur EC2.

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 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 protégés automatiquement. Afin de préserver la contrainte d'immuabilité, les tâches existantes ne sont pas affectées. Pour protéger immédiatement une tâche, vous devez effectuer l'une des actions suivantes :

- Pour les tâches autonomes, arrêtez les tâches, puis démarrez-les.
- Pour les tâches faisant partie d'un service, mettez à jour le service avec l'option « forcer un nouveau déploiement ».

Supprimer la surveillance du temps d'exécution d'un cluster

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 conteneur EC2 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.

Mise à jour GuardDuty de l'agent de sécurité sur vos instances de conteneur EC2

Pour plus d'informations sur la mise à jour de l'agent GuardDuty de sécurité sur vos instances de conteneur EC2, consultez la section Mise à jour de l'agent GuardDuty de sécurité dans le guide de GuardDuty l'utilisateur Amazon.

Supprimer la surveillance du temps d'exécution 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 plus d'informations, consultez Supprimer la surveillance du temps d'exécution d'un cluster.

FAQ sur le dépannage de la 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 conteneur EC2 ?
- Que se passe-t-il lorsqu'il n'existe aucun rôle d'exécution de tâche pour une tâche exécutée sur le cluster ?
- Comment puis-je 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 ?

Dans la console Amazon ECS, les informations se trouvent dans l'onglet Tags de la page détaillée du cluster.

Vous pouvez également exécuter `describe-clusters` avec l'TAGSoption.

L'exemple suivant montre la sortie du cluster par défaut

```
aws ecs describe-clusters --cluster default --include TAGS
```

Sortie

La balise dont la clé est définie sur `GuardDutyManaged` et la valeur définie sur `true` indique que le cluster est configuré pour la surveillance du temps d'exécution.

```
{
    "clusters": [
        {
            "clusterArn": "arn:aws:ecs:us-east-1:1234567890:cluster/default",
            "clusterName": "default",
            "status": "ACTIVE",
            "registeredContainerInstancesCount": 0,
            "runningTasksCount": 1,
            "pendingTasksCount": 0,
            "activeServicesCount": 0,
            "statistics": [],
            "tags": [
                {
                    "key": "GuardDutyManaged",
                    "value": "true"
                }
            ],
            "settings": [],
            "capacityProviders": [],
            "defaultCapacityProviderStrategy": []
        }
    ],
    "failures": []
}
```

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 conteneur EC2 ?

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 puis-je 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 dans ou obtenir un shell vers un conteneur s'exécutant sur une instance Amazon EC2 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), des AWS SDK ou de la 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 stratégies de contrôle d'accès plus strictes et vérifier l'accès aux conteneurs. 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 vérifier quelles tâches ont été exécutées et CloudTrail pour vérifier qui a accédé à un conteneur.

Considérations relatives à l'utilisation d'ECS Exec

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.
- 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 AMI Windows optimisées pour Amazon ECS suivantes (avec la version `1.56` ou ultérieure de l'agent de conteneur) :
    - 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
- 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 instances Amazon EC2, 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 (points de terminaison Amazon VPC) 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 plus d'informations, consultez 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. Pour de plus amples informations sur les conteneurs Nano Server, consultez Nano Server sur le site web Docker.

Conditions préalables à l'utilisation d'ECS Exec

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, consultez 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 1.50.2 ou supérieure de l'agent. Pour plus d'informations, consultez AMI optimisées pour Amazon ECS.
  - 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 à l'intérieur du 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 pour une tâche dont l'ID est task-id.

L'*identifiant de tâche* est l'Amazon Resource Name (ARN) de la tâche.

```
aws ecs execute-command --cluster cluster-name \
    --task task-id \
    --container container-name \
    --interactive \
    --command "/bin/sh"
```

Journalisation et audit à l'aide d'ECS Exec

Activation de la journalisation et de l'audit dans 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 plus d'informations, consultez 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 en envoyant des CloudWatch journaux à Logs à l'aide du pilote de `awslogs` journal configuré dans votre définition de tâche. Si vous souhaitez fournir une configuration personnalisée, l' AWS CLI prend en charge un indicateur `--configuration` pour les commandes `create-cluster` et `update-cluster`. Il est également important de savoir que l'image du conteneur nécessite `script` et doit être installée `cat` pour 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.

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
               "logs:CreateLogStream",
               "logs:DescribeLogStreams",
               "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:region:account-id:log-group:/aws/
ecs/cloudwatch-log-group-name:*"
```

```
        }
    ]
}
```

Amazon S3

L'exemple suivant de stratégie ajoute les autorisations Amazon S3 requises.

```
{
   "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 plus d'informations, consultez Autorisations IAM requises pour ECS Exec.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "kms-key-arn"
        }
    ]
}
```

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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "kms-key-arn"
        }
    ]
}
```

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`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ExecuteCommand",
                "ecs:DescribeTasks"
            ],
         "Resource": [
                "arn:aws:ecs:region:aws-account-id:task/cluster-name/*",
                "arn:aws:ecs:region:aws-account-id:cluster/*"
            ],
            "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`.

```
{
    "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.

```
{
    "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`*.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": [
                "arn:aws:ecs:region:aws-account-id:task/cluster-name/*",
                "arn:aws:ecs:region:aws-account-id:cluster/*"
            ]
        }
    ]
```

```
}
```

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`.

```
{
    "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 concernant 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 .

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
- Bonnes pratiques de sécurité

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
- Autorisations requises pour la console Amazon ECS
- Rôle IAM d'exécution de tâche Amazon ECS
- Rôle IAM de tâche
- Rôle IAM dans l'infrastructure Amazon ECS
- Configuration supplémentaire des rôles IAM pour les tâches Windows
- Rôle IAM d'instance de conteneur Amazon ECS
- Rôle IAM ECS Anywhere
- Rôle CodeDeploy IAM d'Amazon ECS
- Rôle EventBridge IAM d'Amazon ECS
- Autorisations IAM requises pour Service Auto Scaling
- Octroi de l'autorisation de baliser les ressources lors de la création
- Dépannage de l'identité et de l'accès Amazon Elastic Container Service

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 vous-même les demandes, consultez la section Signature des demandes AWS 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 en savoir plus, veuillez consulter Multi-factor authentication (Authentification multifactorielle) dans le Guide de l'utilisateur AWS IAM Identity Center et Utilisation de l'authentification multifactorielle (MFA) dans l'interface AWS 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 root, consultez Tâches nécessitant des informations d'identification d'utilisateur root 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 tels que les 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 de faire pivoter les 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 avoir un groupe nommé IAMAdmins et accorder à ce groupe les autorisations d'administrer des 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 en savoir plus, consultez Quand créer un utilisateur IAM (au lieu d'un rôle) 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. Vous pouvez assumer temporairement un rôle IAM dans le en AWS Management Console changeant de rôle. 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 Utilisation de rôles IAM 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, veuillez consulter la rubrique 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 Différence entre les rôles IAM et les politiques basées sur les ressources 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, une fonction 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, l'action que vous effectuez est susceptible de lancer 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 la politique relative à la transmission de demandes FAS, consultez Sessions de transmission d'accès.

- Fonction du 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 une fonction du 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 instance EC2 et qui envoient des demandes d'API. AWS CLI AWS Cette solution est préférable au stockage des clés d'accès au sein de l'instance EC2. Pour attribuer un AWS rôle à une instance EC2 et le mettre à la disposition de 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 qui s'exécutent sur l'instance EC2 d'obtenir des informations d'identification temporaires. Pour plus d'informations, consultez Utilisation d'un rôle IAM pour accorder des autorisations à des applications s'exécutant sur des instances Amazon EC2 dans le Guide de l'utilisateur IAM.

Pour savoir dans quel cas utiliser des rôles ou des utilisateurs IAM, consultez Quand créer un rôle IAM (au lieu d'un utilisateur) 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 Présentation 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 Groupes d'utilisateurs IAM 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 Création de politiques IAM 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. Des politiques basées sur les ressources sont, par exemple, les politiques de confiance de rôle IAM et des politiques de compartiment Amazon S3. 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 (ACL)

Les listes de contrôle d'accès (ACL) vérifie quels principaux (membres de compte, utilisateurs ou rôles) ont l'autorisation d'accéder à une ressource. Les listes de contrôle d'accès sont similaires aux politiques basées sur les ressources, bien qu'elles n'utilisent pas le format de document de politique JSON.

Amazon S3 et Amazon VPC sont des exemples de services qui prennent en charge les ACL. AWS WAF Pour en savoir plus sur les listes de contrôle d'accès, consultez Présentation des listes 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 des 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 IAM ou rôle IAM). Vous pouvez définir une limite d'autorisations pour une entité. Les autorisations qui en résultent 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 remplace 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 (SCP) — Les SCP sont des 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 fonctions d'une organisation, vous pouvez appliquer les politiques de contrôle de service (SCP) 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 organisations et les SCP, consultez Fonctionnement des SCP dans le Guide de l'utilisateur AWS Organizations .

- 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 la séance obtenue 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 remplace l'autorisation. Pour plus d'informations, consultez Politiques de séance dans le Guide de l'utilisateur IAM.

Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations obtenues 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.

Fonctions IAM que vous pouvez utiliser avec Amazon Elastic Container Service

Fonction 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

Fonction IAM	Prise en charge d'Amazon ECS
Clés de condition d’une politique	Oui
ACL	Non
ABAC (étiquettes dans les politiques)	Oui
Informations d’identification temporaires	Oui
Transmission des sessions d’accès (FAS)	Oui
Fonctions 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 Groupes d’utilisateurs IAM 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, veuillez consulter Création de politiques IAM 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. Des politiques basées sur les ressources sont, par exemple, les politiques de confiance de rôle IAM et des politiques de compartiment Amazon S3. 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 entre comptes 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 Différence entre les rôles IAM et les politiques basées sur une ressource 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 stratégie 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

Prend en charge les ressources de politique	Partielle

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 pour lesquels 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 afficher la liste des types de ressources Amazon ECS et leur ARN, consultez Ressources définies par Amazon Elastic Container Service dans la Référence de l'autorisation de service. 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 leurs ARN 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 le cluster `my-cluster-1` et `my-cluster-2` dans votre instruction, utilisez les ARN suivants :

```
"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 la dernière définition de tâche, utilisez :

```
"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	Cette clé de contexte est formatée selon le schéma `"aws:RequestTag/ tag-key":"tag-value "`, où *`tag-key`* et *`tag-value`* représentent respectivement une paire clé-valeur d'étiquette. 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	Cette clé de contexte est formatée selon le schéma `"aws:ResourceTag/ tag-key":"tag-value "`, où	Chaîne

Clé de condition	Description	Types d'évaluation
	tag-key et *tag-value* représentent respectivement une paire clé-valeur d'étiquette. Vérifie que l'étiquette attachée à la ressource d'identité (utilisateur ou rôle) correspond aux nom et valeur de la clé spécifiée.	
lois : TagKeys	Cette clé de contexte est formatée selon le schéma `"aws:TagKeys":" tag-key"`, où *tag-key* est une liste de clés d'étiquette sans valeur (par exemple, `["Dept","Cost-Center"]`). Vérifie les clés de balise présentes dans une AWS demande.	Chaîne
sec : ResourceTag /$ {} TagKey	Cette clé de contexte est formatée selon le schéma `"ecs:ResourceTag/ tag-key":"tag-value "`, où *tag-key* et *tag-value* représentent respectivement une paire clé-valeur d'étiquette. Vérifie que l'étiquette attachée à la ressource d'identité (utilisateur ou rôle) correspond aux nom et valeur de la clé spécifiée.	Chaîne
ecs:cluster	La clé de contexte a le format `"ecs:cluster":" cluster-arn "` où *cluster-arn* est l'ARN pour le cluster Amazon ECS.	ARN, Null
ecs:container-instances	La clé de contexte est formatée `"ecs:container-instances":" container-instance-arns "` là où *container-instance-arns* se trouvent un ou plusieurs ARN d'instance de conteneur.	ARN, Null

Clé de condition	Description	Types d'évaluation
ecs:container-name	La clé contextuelle est formatée `"ecs:container-name":"` *`container-name`* `"`, où *`container-instance`* est le nom d'un conteneur Amazon ECS défini dans la définition de tâche.	Chaîne
ecs : enable-execute-command	La clé contextuelle est formatée `"ecs:enable-execute-command":"` *`value`*`"` où *`value-`* est « `true` » ou « `false` ».	Chaîne
ecs : enable-service-connect	La clé contextuelle est formatée `"ecs:enable-service-connect":"` *`value`*`"` où *`value`* est `"true"` ou `"false"`.	Chaîne
ecs : enable-ebs-volumes	La clé contextuelle est formatée `"ecs:enable-ebs-volumes":"` *`value`*`"` où *`value`* est `"true"` ou `"false"`.	Chaîne
ecs:namespace	La clé de contexte est formatée `"ecs:namespace":"` *`namespace-arn`* `"` où *`namespace-arn`* est l'ARN pour l'espace de nom AWS Cloud Map .	ARN, Null
ecs:service	La clé de contexte a le format `"ecs:service":"` *`service-arn`* `"` où *`service-arn`* est l'ARN pour le service Amazon ECS service.	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:account-setting	La clé de contexte est formatée `"ecs:account-setting":"` *`account-setting`* `"`, où *`account-setting`* est le nom d'un paramètre de compte Amazon ECS.	Chaîne

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 (ACL) dans Amazon ECS

Prend en charge les listes ACL	Non

Les listes de contrôle d’accès (ACL) vérifient quels principaux (membres de compte, utilisateurs ou rôles) ont l’autorisation d’accéder à une ressource. Les listes de contrôle d’accès 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 combinaison action/ressource.

Prend en charge ABAC (étiquettes 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 l'ABAC, consultez Qu'est-ce que le contrôle d'accès basé sur les attributs (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 basé sur les 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 Changement de rôle (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 transmissions de sessions d'accès (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, l'action que vous effectuez est susceptible de lancer 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 fonctions de service	Oui

Une fonction du service est un rôle IAM qu'un service endosse pour accomplir des actions en votre nom. Un administrateur IAM peut créer, modifier et supprimer une fonction du 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 à un service.	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 doit 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 dans l'onglet JSON 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 format des ARN pour chacun des types de ressources, consultez Actions, ressources et clés de condition pour Amazon Elastic Container Service dans la Référence de l'autorisation de service.

Rubriques

- Bonnes pratiques en matière de politiques
- Autorisation accordée aux utilisateurs pour afficher leurs propres autorisations
- Exemples de cluster
- Exemples d'instance de conteneur
- Exemples de définition de tâche
- Exemple d'exécution de tâche
- Exemple de démarrage de tâche
- Exemples d'affichage et de description de tâche
- Exemple de création de service
- Exemple de mise à jour du service
- Description des services Amazon ECS basée sur des balises
- Exemple de refus de remplacement de l'espace de noms Service Connect

Bonnes pratiques en matière de politiques

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 de plus amples informations, consultez Politiques gérées AWS ou Politiques gérées AWS pour les activités professionnelles dans le Guide de l'utilisateur IAM.
- Accorder 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 de IAM pour appliquer des autorisations, consultez Politiques et autorisations dans IAM dans le Guide de l'utilisateur IAM.

- Utiliser 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 IAM Access Analyzer pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles - IAM Access Analyzer 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 de plus amples informations, consultez Validation de politique 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 le MFA lorsque des opérations d'API sont appelées, ajoutez des conditions MFA à vos politiques. Pour de plus amples informations, consultez Configuration de l'accès aux API protégé par 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.

Autorisation accordée aux utilisateurs pour afficher 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 cluster

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 actions `DescribeClusters` et `DeleteCluster` acceptent les ARN de cluster 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'instance de conteneur

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 tâche

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 d'exécution de tâche

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 démarrage de tâche

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>:*"]
        }
    ]
}
```

Exemples d'affichage et de description de tâche

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>"]
        }
    ]
}
```

Exemple de création de service

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

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.

```
{
    "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 `0wner=richard-roe` ou `owner=richard-roe`. Dans le cas contraire, l'utilisateur se voit refuser l'accès. La clé de condition d'étiquette `0wner` correspond à la fois à `0wner` 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 refus de remplacement de l'espace de noms 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 stratégies gérées et relations d'approbation que vous pouvez attacher à des utilisateurs, groupes, rôles, des instances Amazon EC2 et des tâches Amazon ECS qui autorisent 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.

Détails de l’autorisation

La stratégie IAM gérée par `AmazonECS_FullAccess` doit inclure les autorisations suivantes. 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.

- `ecs`— Permet aux principaux un accès complet à toutes les opérations de l'API Amazon ECS.
- `application-autoscaling` : permet aux mandataires de créer, de décrire et de gérer des ressources Application Auto Scaling. Ceci est requis lors de l'activation de la scalabilité automatique du service pour vos services Amazon ECS.
- `appmesh` : permet aux mandataires de répertorier les maillages de service et les nœuds virtuels App Mesh et de décrire les nœuds virtuels App Mesh. Ceci est requis lors de l'intégration de vos services Amazon ECS service avec App Mesh.
- `autoscaling` : permet aux mandataires de créer, de décrire et de gérer des ressources 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 du cluster.
- `cloudformation`— Permet aux directeurs de créer et de gérer des AWS CloudFormation piles. 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.
- `cloudwatch`— Permet aux directeurs de créer, de gérer et de décrire les CloudWatch alarmes Amazon.
- `codedeploy`— Permet aux responsables de créer et de gérer des déploiements d'applications et de visualiser leurs configurations, leurs révisions et leurs cibles de déploiement.
- `sns` : permet aux mandataires d'obtenir une liste de rubriques Amazon SNS.
- `lambda` : permet aux mandataires d'afficher une liste de fonctions AWS Lambda et les configurations spécifiques à leur version.
- `ec2`— Permet aux principaux d'exécuter des instances Amazon EC2 et de créer et gérer des itinéraires, des tables de routage, des passerelles Internet, des groupes de lancement, des groupes de sécurité, des clouds privés virtuels, des flottes ponctuelles et des sous-réseaux.

- `elasticloadbalancing` : permet aux mandataires de créer, décrire et supprimer des équilibreurs de charge Elastic Load Balancing. Les principaux seront également en mesure d'ajouter des balises aux groupes cibles, écouteurs et règles d'écouteurs récemment créés pour les équilibreurs de charge.
- `events`— Permet aux directeurs de créer, de gérer et de supprimer les EventBridge règles Amazon et leurs cibles.
- `iam` : permet aux mandataires de répertorier les rôles IAM et les stratégies qui leur sont attachées. Les mandataires peuvent également répertorier les profils d'instance disponibles pour vos instances Amazon EC2.
- `logs`— Permet aux principaux de créer et de décrire des groupes de CloudWatch journaux Amazon Logs. Les mandataires peuvent également répertorier les événements de journal pour ces groupes de journaux.
- `route53` : permet aux mandataires de créer, de gérer et de supprimer des zones hébergées Amazon Route 53. Les mandataires peuvent également consulter la configuration et les informations de surveillance de l'état d'Amazon Route 53. Pour plus d'informations sur les zones hébergées, veuillez consulter Utilisation des zones hébergées.
- `servicediscovery`— Permet aux principaux de créer, de gérer et de supprimer des AWS Cloud Map services et de créer des espaces de noms DNS privés.

Voici un exemple de politique `AmazonECS_FullAccess`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:DeleteScalingPolicy",
                "application-autoscaling:DeregisterScalableTarget",
                "application-autoscaling:DescribeScalableTargets",
                "application-autoscaling:DescribeScalingActivities",
                "application-autoscaling:DescribeScalingPolicies",
                "application-autoscaling:PutScalingPolicy",
                "application-autoscaling:RegisterScalableTarget",
                "appmesh:DescribeVirtualGateway",
                "appmesh:DescribeVirtualNode",
                "appmesh:ListMeshes",
                "appmesh:ListVirtualGateways",
```

```
                "appmesh:ListVirtualNodes",
                "autoscaling:CreateAutoScalingGroup",
                "autoscaling:CreateLaunchConfiguration",
                "autoscaling:DeleteAutoScalingGroup",
                "autoscaling:DeleteLaunchConfiguration",
                "autoscaling:Describe*",
                "autoscaling:UpdateAutoScalingGroup",
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:PutMetricAlarm",
                "codedeploy:BatchGetApplicationRevisions",
                "codedeploy:BatchGetApplications",
                "codedeploy:BatchGetDeploymentGroups",
                "codedeploy:BatchGetDeployments",
                "codedeploy:ContinueDeployment",
                "codedeploy:CreateApplication",
                "codedeploy:CreateDeployment",
                "codedeploy:CreateDeploymentGroup",
                "codedeploy:GetApplication",
                "codedeploy:GetApplicationRevision",
                "codedeploy:GetDeployment",
                "codedeploy:GetDeploymentConfig",
                "codedeploy:GetDeploymentGroup",
                "codedeploy:GetDeploymentTarget",
                "codedeploy:ListApplicationRevisions",
                "codedeploy:ListApplications",
                "codedeploy:ListDeploymentConfigs",
                "codedeploy:ListDeploymentGroups",
                "codedeploy:ListDeployments",
                "codedeploy:ListDeploymentTargets",
                "codedeploy:RegisterApplicationRevision",
                "codedeploy:StopDeployment",
                "ec2:AssociateRouteTable",
                "ec2:AttachInternetGateway",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CancelSpotFleetRequests",
                "ec2:CreateInternetGateway",
                "ec2:CreateLaunchTemplate",
                "ec2:CreateRoute",
```

```
                "ec2:CreateRouteTable",
                "ec2:CreateSecurityGroup",
                "ec2:CreateSubnet",
                "ec2:CreateVpc",
                "ec2:DeleteLaunchTemplate",
                "ec2:DeleteSubnet",
                "ec2:DeleteVpc",
                "ec2:Describe*",
                "ec2:DetachInternetGateway",
                "ec2:DisassociateRouteTable",
                "ec2:ModifySubnetAttribute",
                "ec2:ModifyVpcAttribute",
                "ec2:RequestSpotFleet",
                "ec2:RunInstances",
                "ecs:*",
                "elasticfilesystem:DescribeAccessPoints",
                "elasticfilesystem:DescribeFileSystems",
                "elasticloadbalancing:CreateListener",
                "elasticloadbalancing:CreateLoadBalancer",
                "elasticloadbalancing:CreateRule",
                "elasticloadbalancing:CreateTargetGroup",
                "elasticloadbalancing:DeleteListener",
                "elasticloadbalancing:DeleteLoadBalancer",
                "elasticloadbalancing:DeleteRule",
                "elasticloadbalancing:DeleteTargetGroup",
                "elasticloadbalancing:DescribeListeners",
                "elasticloadbalancing:DescribeLoadBalancers",
                "elasticloadbalancing:DescribeRules",
                "elasticloadbalancing:DescribeTargetGroups",
                "events:DeleteRule",
                "events:DescribeRule",
                "events:ListRuleNamesByTarget",
                "events:ListTargetsByRule",
                "events:PutRule",
                "events:PutTargets",
                "events:RemoveTargets",
                "fsx:DescribeFileSystems",
                "iam:ListAttachedRolePolicies",
                "iam:ListInstanceProfiles",
                "iam:ListRoles",
                "lambda:ListFunctions",
                "logs:CreateLogGroup",
                "logs:DescribeLogGroups",
                "logs:FilterLogEvents",
```

```
                "route53:CreateHostedZone",
                "route53:DeleteHostedZone",
                "route53:GetHealthCheck",
                "route53:GetHostedZone",
                "route53:ListHostedZonesByName",
                "servicediscovery:CreatePrivateDnsNamespace",
                "servicediscovery:CreateService",
                "servicediscovery:DeleteService",
                "servicediscovery:GetNamespace",
                "servicediscovery:GetOperation",
                "servicediscovery:GetService",
                "servicediscovery:ListNamespaces",
                "servicediscovery:ListServices",
                "servicediscovery:UpdateService",
                "sns:ListTopics"
            ],
            "Resource": ["*"]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameter",
                "ssm:GetParameters",
                "ssm:GetParametersByPath"
            ],
            "Resource": "arn:aws:ssm:*:*:parameter/aws/service/ecs*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteInternetGateway",
                "ec2:DeleteRoute",
                "ec2:DeleteRouteTable",
                "ec2:DeleteSecurityGroup"
            ],
            "Resource": ["*"],
            "Condition": {
                "StringLike": {"ec2:ResourceTag/aws:cloudformation:stack-name":
"EC2ContainerService-*"}
            }
        },
        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
```

```
            "Resource": ["*"],
            "Condition": {
                "StringLike": {"iam:PassedToService": "ecs-tasks.amazonaws.com"}
            }
        },
        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource": ["arn:aws:iam::*:role/ecsInstanceRole*"],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com",
                        "ec2.amazonaws.com.cn"
                    ]
                }
            }
        },
        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource": ["arn:aws:iam::*:role/ecsAutoscaleRole*"],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": [
                        "application-autoscaling.amazonaws.com",
                        "application-autoscaling.amazonaws.com.cn"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": [
                        "autoscaling.amazonaws.com",
                        "ecs.amazonaws.com",
                        "ecs.application-autoscaling.amazonaws.com",
                        "spot.amazonaws.com",
                        "spotfleet.amazonaws.com"
                    ]
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": ["elasticloadbalancing:AddTags"],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "elasticloadbalancing:CreateAction": [
                        "CreateTargetGroup",
                        "CreateRule",
                        "CreateListener",
                        "CreateLoadBalancer"
                    ]
                }
            }
        }
    ]
}
```

AmazonECS InfrastructureRolePolicyForVolumes

La politique IAM `AmazonECSInfrastructureRolePolicyForVolumes` gérée accorde les autorisations nécessaires à Amazon ECS pour effectuer des appels d' AWS 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.

Détails de l’autorisation

La stratégie IAM gérée par `AmazonECSInfrastructureRolePolicyForVolumes` doit inclure les autorisations suivantes. En suivant les conseils de sécurité standard relatifs à l'octroi du moindre privilège, vous pouvez utiliser la politique `AmazonECSInfrastructureRolePolicyForVolumes` gérée comme modèle pour créer votre propre politique personnalisée qui inclut uniquement les autorisations dont vous avez besoin.

- `ec2:CreateVolume`— Permet à un principal de créer un volume Amazon EBS si et uniquement s'il est étiqueté avec les `AmazonECSManaged` balises `AmazonECSCreated` et. Cette autorisation

est requise pour créer des volumes Amazon EBS attachés aux tâches Amazon ECS et pour minimiser les autorisations accordées à Amazon ECS par cette politique.

- `ec2:CreateTags`— Permet à un principal d'ajouter des balises à un volume Amazon EBS dans le cadre de`ec2:CreateVolume`. Cette autorisation est requise par Amazon ECS pour ajouter des balises spécifiées par le client aux volumes Amazon EBS créés en votre nom.
- `ec2:AttachVolume`— Permet à un principal d'associer un volume Amazon EBS à une instance Amazon EC2. Cette autorisation est requise par Amazon ECS pour associer des volumes Amazon EBS à l'instance Amazon EC2 hébergeant la tâche Amazon ECS associée.
- `ec2:DescribeVolume`— Permet au principal de récupérer des informations sur les volumes Amazon EBS. Cette autorisation est requise pour gérer le cycle de vie des volumes Amazon EBS.
- `ec2:DescribeAvailabilityZones`— Permet à un mandant de récupérer des informations sur les zones de disponibilité de votre compte. Cela est nécessaire pour gérer le cycle de vie des volumes EBS.
- `ec2:DetachVolume`— Permet à un principal de détacher un volume Amazon EBS d'une instance Amazon EC2. Cette autorisation est requise par Amazon ECS pour détacher le volume Amazon EBS de l'instance Amazon EC2 qui héberge la tâche Amazon ECS associée lorsque la tâche se termine.
- `ec2:DeleteVolume`— Permet au principal de supprimer un volume Amazon EBS. Cette autorisation est requise par Amazon ECS pour supprimer les volumes Amazon EBS qui ne sont plus utilisés par la tâche Amazon ECS.
- `ec2:DeleteTags`— Permet au principal de supprimer le `AmazonECSManaged` tag d'un volume Amazon EBS. Cette autorisation est requise par Amazon ECS pour supprimer l'accès à un volume Amazon EBS une fois que celui-ci n'est plus associé à une charge de travail Amazon ECS. Cela ne s'applique que lorsqu'un volume Amazon EBS n'est pas supprimé après l'arrêt de la tâche.

Voici un exemple de politique `AmazonECSInfrastructureRolePolicyForVolumes`.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "CreateEBSManagedVolume",
   "Effect": "Allow",
   "Action": "ec2:CreateVolume",
   "Resource": "arn:aws:ec2:*:*:volume/*",
   "Condition": {
```

```
    "ArnLike": {
     "aws:RequestTag/AmazonECSCreated": "arn:aws:ecs:*:*:task/*"
    },
    "StringEquals": {
     "aws:RequestTag/AmazonECSManaged": "true"
    }
   }
  },
  {
   "Sid": "TagOnCreateVolume",
   "Effect": "Allow",
   "Action": "ec2:CreateTags",
   "Resource": "arn:aws:ec2:*:*:volume/*",
   "Condition": {
    "ArnLike": {
     "aws:RequestTag/AmazonECSCreated": "arn:aws:ecs:*:*:task/*"
    },
    "StringEquals": {
     "ec2:CreateAction": "CreateVolume",
     "aws:RequestTag/AmazonECSManaged": "true"
    }
   }
  },
  {
   "Sid": "DescribeVolumesForLifecycle",
   "Effect": "Allow",
   "Action": [
    "ec2:DescribeVolumes",
    "ec2:DescribeAvailabilityZones"
   ],
   "Resource": "*"
  },
  {
   "Sid": "ManageEBSVolumeLifecycle",
   "Effect": "Allow",
   "Action": [
    "ec2:AttachVolume",
    "ec2:DetachVolume"
   ],
   "Resource": "arn:aws:ec2:*:*:volume/*",
   "Condition": {
    "StringEquals": {
     "aws:ResourceTag/AmazonECSManaged": "true"
    }
```

```
    }
   },
   {
    "Sid": "ManageVolumeAttachmentsForEC2",
    "Effect": "Allow",
    "Action": [
     "ec2:AttachVolume",
     "ec2:DetachVolume"
    ],
    "Resource": "arn:aws:ec2:*:*:instance/*"
   },
   {
    "Sid": "DeleteEBSManagedVolume",
    "Effect": "Allow",
    "Action": "ec2:DeleteVolume",
    "Resource": "arn:aws:ec2:*:*:volume/*",
    "Condition": {
     "ArnLike": {
      "aws:ResourceTag/AmazonECSCreated": "arn:aws:ecs:*:*:task/*"
     },
     "StringEquals": {
      "aws:ResourceTag/AmazonECSManaged": "true"
     }
    }
   }
  ]
}
```

Rôle Amazon EC2 EC2 ContainerServicefor

Amazon ECS associe à cette politique une fonction de service qui permet à Amazon ECS d'effectuer des actions en votre nom sur des instances Amazon EC2 ou des instances externes.

Cette politique accorde des autorisations administratives qui permettent aux instances de conteneur Amazon ECS de passer des appels AWS en votre nom. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.

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. Pour plus d'informations, consultez Détails de l'autorisation.
- 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 plus d'informations, consultez Rôle IAM de tâche.

  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 tâche 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 plus d'informations, consultez 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
```

Détails de l’autorisation

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` : permet à un principal de décrire les balises associées à une instance Amazon EC2. Cette autorisation est utilisée par l'agent de conteneur Amazon ECS pour prendre en charge la propagation des balises de ressources. Pour plus d’informations, consultez Comment les ressources sont étiquetées.
- `ecs:CreateCluster` : permet au 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` : permet au principal de désenregistrer une instance de conteneur Amazon ECS à partir 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` : cette action renvoie les points de terminaison utilisés par l'agent de conteneur Amazon ECS pour rechercher les mises à jour.
- `ecs:Poll` : permet à l'agent de conteneur Amazon ECS de communiquer avec le plan de contrôle Amazon ECS pour signaler les changements d'état de tâche.
- `ecs:RegisterContainerInstance` : permet au principal d'enregistrer une instance de conteneur avec un cluster. Cette autorisation est utilisée par l'agent de conteneur Amazon ECS pour enregistrer l'instance Amazon EC2 auprès d'un cluster et pour prendre en charge la propagation des balises de ressources.
- `ecs:StartTelemetrySession` : permet à l'agent de conteneur Amazon ECS de communiquer avec le plan de contrôle Amazon ECS pour signaler les informations et métriques d'état de chaque conteneur et tâche.
- `ecs:TagResource` : 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. Pour plus d'informations, consultez Drainage des instances Spot.
- `ecs:Submit*` : cela inclut les actions d'API `SubmitAttachmentStateChanges`, `SubmitContainerStateChange` et `SubmitTaskStateChange`. 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` : permet au 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` : lorsqu'une image de conteneur est transférée vers un référentiel privé Amazon ECR, chaque couche d'image est examinée afin de vérifier si elle a déjà été transférée. Le cas échéant, le calque d'image est ignoré.
- `ecr:GetDownloadUrlForLayer` : 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` : lorsqu'une image de conteneur est extraite d'un référentiel privé Amazon ECR, cette API est appelée une fois pour récupérer le manifeste d'image.
- `logs:CreateLogStream`— Permet à un principal de créer un flux de CloudWatch journaux pour un groupe de journaux spécifié.
- `logs:PutLogEvents` : permet au principal de charger un lot d'événements de journaux dans un flux de journaux spécifié.

Voici un exemple de stratégie `AmazonEC2ContainerServiceforEC2Role`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeTags",
                "ecs:CreateCluster",
```

```
                "ecs:DeregisterContainerInstance",
                "ecs:DiscoverPollEndpoint",
                "ecs:Poll",
                "ecs:RegisterContainerInstance",
                "ecs:StartTelemetrySession",
                "ecs:UpdateContainerInstancesState",
                "ecs:Submit*",
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ecs:TagResource",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:CreateAction": [
                        "CreateCluster",
                        "RegisterContainerInstance"
                    ]
                }
            }
        }
    ]
}
```

Amazon EC2 ContainerServiceEventsRole

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 plus d’informations, consultez Rôle EventBridge IAM d'Amazon ECS.

Détails de l’autorisation

Cette politique inclut les autorisations suivantes.

- `ecs`— Permet au principal d'un service d'appeler l' RunTask API Amazon ECS. Permet au principal d'un service d'ajouter des tags (`TagResource`) lorsqu'il appelle l' RunTask API Amazon ECS.
- `iam` : permet de transmettre n'importe quel rôle de service IAM à toutes les tâches Amazon ECS.

Voici un exemple de stratégie `AmazonEC2ContainerServiceEventsRole`.

```
{
    "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"]
                }
            }
        }
    ]
}
```

AmazonECS TaskExecutionRolePolicy

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 plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Détails de l'autorisation

La stratégie IAM gérée par `AmazonECSTaskExecutionRolePolicy` doit inclure les autorisations suivantes. Conformément aux conseils de sécurité standard pour accorder le moindre privilège, la stratégie gérée par `AmazonECSTaskExecutionRolePolicy` peut être utilisée comme guide. Si l'une des autorisations qui sont 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.

- `ecr:GetAuthorizationToken` : permet au 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` : lorsqu'une image de conteneur est transférée vers un référentiel privé Amazon ECR, chaque couche d'image est examinée afin de vérifier si elle a déjà été transférée. Si elle a été transmise, le calque d'image est ignoré.
- `ecr:GetDownloadUrlForLayer` : 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` : lorsqu'une image de conteneur est extraite d'un référentiel privé Amazon ECR, cette API est appelée une fois pour récupérer le manifeste d'image.
- `logs:CreateLogStream`— Permet à un principal de créer un flux de CloudWatch journaux pour un groupe de journaux spécifié.
- `logs:PutLogEvents` : permet au principal de charger un lot d'événements de journaux dans un flux de journaux spécifié.

Voici un exemple de stratégie `AmazonECSTaskExecutionRolePolicy`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
    ]
}
```

AmazonECS ServiceRolePolicy

La politique IAM gérée `AmazonECSServiceRolePolicy` permet à Amazon Elastic Container Service de gérer votre cluster. Cette stratégie peut être ajoutée à votre rôle IAM d'exécution de tâche. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Détails de l'autorisation

La stratégie IAM gérée par `AmazonECSServiceRolePolicy` doit inclure les autorisations suivantes. Conformément aux conseils de sécurité standard pour accorder le moindre privilège, la stratégie gérée par `AmazonECSServiceRolePolicy` peut être utilisée comme guide. Si l'une des autorisations qui sont 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.

- `autoscaling` : permet aux mandataires de créer, de décrire et de gérer des ressources 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 du cluster.
- `autoscaling-plans` : permet aux principaux de créer, supprimer et décrire des plans de mise à l'échelle automatique.
- `cloudwatch`— Permet aux directeurs de créer, de gérer et de décrire les CloudWatch alarmes Amazon.
- `ec2`— Permet aux principaux de s'exécuter sur des instances Amazon EC2 et de créer et de gérer des interfaces réseau et des balises.
- `elasticloadbalancing` : permet aux mandataires de créer, décrire et supprimer des équilibreurs de charge Elastic Load Balancing. Les directeurs seront également en mesure d'ajouter et de décrire des groupes cibles.

- `logs`— Permet aux principaux de créer et de décrire des groupes de CloudWatch journaux Amazon Logs. Les mandataires peuvent également répertorier les événements de journal pour ces groupes de journaux.
- `route53` : permet aux mandataires de créer, de gérer et de supprimer des zones hébergées Amazon Route 53. Les mandataires peuvent également consulter la configuration et les informations de surveillance de l'état d'Amazon Route 53. Pour plus d'informations sur les zones hébergées, veuillez consulter Utilisation des zones hébergées.
- `servicediscovery`— Permet aux principaux de créer, de gérer et de supprimer des AWS Cloud Map services et de créer des espaces de noms DNS privés.
- `events`— Permet aux directeurs de créer, de gérer et de supprimer les EventBridge règles Amazon et leurs cibles.

Voici un exemple de politique `AmazonECSServiceRolePolicy`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ECSTaskManagement",
            "Effect": "Allow",
            "Action": [
                "ec2:AttachNetworkInterface",
                "ec2:CreateNetworkInterface",
                "ec2:CreateNetworkInterfacePermission",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteNetworkInterfacePermission",
                "ec2:Describe*",
                "ec2:DetachNetworkInterface",
                "elasticloadbalancing:DeregisterInstancesFromLoadBalancer",
                "elasticloadbalancing:DeregisterTargets",
                "elasticloadbalancing:Describe*",
                "elasticloadbalancing:RegisterInstancesWithLoadBalancer",
                "elasticloadbalancing:RegisterTargets",
                "route53:ChangeResourceRecordSets",
                "route53:CreateHealthCheck",
                "route53:DeleteHealthCheck",
                "route53:Get*",
                "route53:List*",
                "route53:UpdateHealthCheck",
                "servicediscovery:DeregisterInstance",
```

```
                "servicediscovery:Get*",
                "servicediscovery:List*",
                "servicediscovery:RegisterInstance",
                "servicediscovery:UpdateInstanceCustomHealthStatus"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AutoScaling",
            "Effect": "Allow",
            "Action": [
                "autoscaling:Describe*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AutoScalingManagement",
            "Effect": "Allow",
            "Action": [
                "autoscaling:DeletePolicy",
                "autoscaling:PutScalingPolicy",
                "autoscaling:SetInstanceProtection",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:PutLifecycleHook",
                "autoscaling:DeleteLifecycleHook",
                "autoscaling:CompleteLifecycleAction",
                "autoscaling:RecordLifecycleActionHeartbeat"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "autoscaling:ResourceTag/AmazonECSManaged": "false"
                }
            }
        },
        {
            "Sid": "AutoScalingPlanManagement",
            "Effect": "Allow",
            "Action": [
                "autoscaling-plans:CreateScalingPlan",
                "autoscaling-plans:DeleteScalingPlan",
                "autoscaling-plans:DescribeScalingPlans",
                "autoscaling-plans:DescribeScalingPlanResources"
            ],
```

```
            "Resource": "*"
        },
        {
            "Sid": "EventBridge",
            "Effect": "Allow",
            "Action": [
                "events:DescribeRule",
                "events:ListTargetsByRule"
            ],
            "Resource": "arn:aws:events:*:*:rule/ecs-managed-*"
        },
        {
            "Sid": "EventBridgeRuleManagement",
            "Effect": "Allow",
            "Action": [
                "events:PutRule",
                "events:PutTargets"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "events:ManagedBy": "ecs.amazonaws.com"
                }
            }
        },
        {
            "Sid": "CWAlarmManagement",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm"
            ],
            "Resource": "arn:aws:cloudwatch:*:*:alarm:*"
        },
        {
            "Sid": "ECSTagging",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*:*:network-interface/*"
        },
        {
```

```
            "Sid": "CWLogGroupManagement",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:DescribeLogGroups",
                "logs:PutRetentionPolicy"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/ecs/*"
        },
        {
            "Sid": "CWLogStreamManagement",
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:DescribeLogStreams",
                "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:/aws/ecs/*:log-stream:*"
        },
        {
            "Sid": "ExecuteCommandSessionManagement",
            "Effect": "Allow",
            "Action": [
                "ssm:DescribeSessions"
            ],
            "Resource": "*"
        },
        {
            "Sid": "ExecuteCommand",
            "Effect": "Allow",
            "Action": [
                "ssm:StartSession"
            ],
            "Resource": [
                "arn:aws:ecs:*:*:task/*",
                "arn:aws:ssm:*:*:document/AmazonECS-ExecuteInteractiveCommand"
            ]
        },
        {
            "Sid": "CloudMapResourceCreation",
            "Effect": "Allow",
            "Action": [
                "servicediscovery:CreateHttpNamespace",
                "servicediscovery:CreateService"
```

```
            ],
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "AmazonECSManaged"
                    ]
                }
            }
        },
        {
            "Sid": "CloudMapResourceTagging",
            "Effect": "Allow",
            "Action": "servicediscovery:TagResource",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aws:RequestTag/AmazonECSManaged": "*"
                }
            }
        },
        {
            "Sid": "CloudMapResourceDeletion",
            "Effect": "Allow",
            "Action": [
                "servicediscovery:DeleteService"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/AmazonECSManaged": "false"
                }
            }
        },
        {
            "Sid": "CloudMapResourceDiscovery",
            "Effect": "Allow",
            "Action": [
                "servicediscovery:DiscoverInstances",
                "servicediscovery:DiscoverInstancesRevision"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecur

Fournit un accès administratif à AWS Private Certificate Authority Secrets Manager et AWS aux autres services nécessaires pour gérer les fonctionnalités TLS d'Amazon ECS Service Connect en votre nom.

Détails de l’autorisation

La stratégie IAM gérée par `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` doit inclure les autorisations suivantes. Conformément aux conseils de sécurité standard pour accorder le moindre privilège, la stratégie gérée par `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` peut être utilisée comme guide. Si l'une des autorisations qui sont 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.

- `secretsmanager:CreateSecret`— Permet au principal de créer le secret. C'est obligatoire pour Service Connect TLS, Amazon ECS conserve la clé privée du client secrète dans le Secrets Manager du client.
- `secretsmanager:TagResource`— Permet au principal d'attacher une étiquette au secret créé. Il est nécessaire pour Service Connect TLS, car Amazon ECS crée le secret pour le compte du client et associe une balise à la ressource. Ces balises permettent au client d'identifier plus facilement le secret géré et de limiter les actions relatives à ces secrets.
- `secretsmanager:DescribeSecret`— Autoriser le principal à décrire le secret et à récupérer le stade de version actuel. Amazon ECS doit effectuer la rotation des matériaux Amazon ECS Service Connect TLS.
- `secretsmanager:UpdateSecret`— Autoriser le principal à mettre à jour le secret. Amazon ECS doit effectuer la rotation des matériaux Amazon ECS Service Connect TLS et mettre à jour le secret avec de nouveaux documents.
- `secretsmanager:GetSecretValue`— Autorise le principal à obtenir la valeur secrète. Amazon ECS doit effectuer la rotation des matériaux Amazon ECS Service Connect TLS.
- `secretsmanager:PutSecretValue`— Autorise le principal à saisir la valeur secrète. Amazon ECS doit effectuer la rotation des matériaux Amazon ECS Service Connect TLS.

- `secretsmanager:UpdateSecretVersionStage`— Autoriser le principal à mettre à jour l'étape de la version secrète. Amazon ECS doit effectuer la rotation des matériaux Amazon ECS Service Connect TLS.
- `acm-pca:IssueCertificate`— Autoriser le principal IssueCertificate `End entity certificate` à appeler Amazon ECS Service Connect TLS. ECS devait générer un certificat pour le service en amont du client.
- `acm-pca:GetCertificate`— Autoriser le principal GetCertificate `End entity certificate` à appeler Amazon ECS Service Connect TLS.
- `acm-pca:GetCertificateAuthorityCertificate`— Autoriser le principal à obtenir le certificat des autorités de certification. Il est nécessaire pour le protocole TLS Amazon ECS Service Connect afin que le service en aval du client puisse faire confiance au certificat d'entité en amont.
- `acm-pca:DescribeCertificateAuthority`— Autoriser le principal à obtenir des informations sur l'autorité de certification. Amazon ECS Service Connect TLS doit réutiliser des informations telles que l'algorithme de signature pour créer le CSR (Certificate Signing Request).

Voici un exemple de stratégie `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateSecret",
            "Effect": "Allow",
            "Action": "secretsmanager:CreateSecret",
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*",
            "Condition": {
                "ArnLike": {
                    "aws:RequestTag/AmazonECSCreated": [
                        "arn:aws:ecs:*:*:service/*/*",
                        "arn:aws:ecs:*:*:task-set/*/*"
                    ]
                },
                "StringEquals": {
                    "aws:RequestTag/AmazonECSManaged": "true",
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
            }
        },
```

```
        {
            "Sid": "TagOnCreateSecret",
            "Effect": "Allow",
            "Action": "secretsmanager:TagResource",
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*",
            "Condition": {
                "ArnLike": {
                     "aws:RequestTag/AmazonECSCreated": [
                        "arn:aws:ecs:*:*:service/*/*",
                        "arn:aws:ecs:*:*:task-set/*/*"
                    ]
                },
                "StringEquals": {
                    "aws:RequestTag/AmazonECSManaged": "true",
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
            }
        },
        {
            "Sid": "RotateTLSCertificateSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:UpdateSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:DeleteSecret",
                "secretsmanager:RotateSecret",
                "secretsmanager:UpdateSecretVersionStage"
            ],
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:ResourceTag/aws:secretsmanager:owningService":
 "ecs-sc",
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
            }
        },
        {
            "Sid": "ManagePrivateCertificateAuthority",
            "Effect": "Allow",
            "Action": [
                "acm-pca:GetCertificate",
```

```
                "acm-pca:GetCertificateAuthorityCertificate",
                "acm-pca:DescribeCertificateAuthority"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/AmazonECSManaged": "true"
                }
            }
        },
        {
            "Sid": "ManagePrivateCertificateAuthorityForIssuingEndEntityCertificate",
            "Effect": "Allow",
            "Action": [
                "acm-pca:IssueCertificate"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/AmazonECSManaged": "true",
                    "acm-pca:TemplateArn":"arn:aws:acm-pca:::template/
EndEntityCertificate/V1"
                }
            }
        }
    ]
}
```

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.

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, voir Créer un rôle de service pour CodeDeploy dans le Guide de AWS CodeDeploy l'utilisateur.

`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, voir Créer un rôle de service pour CodeDeploy dans le Guide de AWS CodeDeploy l'utilisateur.

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 une nouvelle politique AmazonECS InfrastructureRolePolicyForServiceConnectTransport LayerSecurity	Ajout d'une nouvelle InfrastructureRolePolicyForServiceConnectTransport LayerSecurity politique AmazonECS 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 | AmazonECS InfrastructureRolePolicyForVolumes	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	11 janvier 2024

Modification	Description	Date
	EBS associés aux charges de travail Amazon ECS.	
Ajouter des autorisations à AmazonECS ServiceRolePolicy	La politique IAM `AmazonECSServiceRolePolicy` gérée a été mise à jour avec de nouvelles `events` autorisations `autoscaling` et `autoscaling-plans` autorisations supplémentaires.	4 décembre 2023
Ajouter des autorisations à AmazonEC2 ContainerServiceEventsRole	La politique IAM `AmazonECSServiceRolePolicy` gérée a été mise à jour pour autoriser l'accès au fonctionnement de l' AWS Cloud Map `DiscoverInstancesRevision` API.	4 octobre 2023
Ajouter des autorisations à Amazon EC2 EC2RoleContainerServicefor	La `AmazonEC2ContainerServiceforEC2Role` politique a été modifiée pour ajouter l'`ecs:TagResource` autorisation, qui inclut une condition qui limite l'autorisation uniquement aux clusters nouvellement créés et aux instances de conteneur enregistrées.	06 mars 2023

Modification	Description	Date
Ajouter des autorisations à the section called “Amazon ECS_ FullAccess”	La `AmazonECS_FullAccess` politique a été modifiée pour ajouter l'`elasticloadbalancing:AddTags` autorisation, qui inclut une condition qui limite l'autorisation uniquement aux équilibreurs de charge, aux groupes cibles, aux règles et aux écouteurs créés récemment. Cette autorisation n'autorise pas l'ajout de balises à des ressources Elastic Load Balancing déjà créées.	4 janvier 2023
Amazon ECS a commencé à assurer le suivi des modifications	Amazon ECS a commencé à suivre les modifications apportées AWS à ses politiques 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 plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez 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 politique AmazonEC2. ContainerServiceFullAccess

    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 prenant en charge les rôles liés à un service, consultez les AWS services opérationnels avec IAM et recherchez les services présentant la mention Yes (Oui) dans la colonne Service-linked roles (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 au service nommé. AWSServiceRoleForECS

Le rôle AWSServiceRoleForECS lié à un service fait confiance aux services suivants pour assumer le rôle :

- `ecs.amazonaws.com`

La politique d'autorisations de rôle nommée AmazonECS ServiceRolePolicy 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 instances Amazon EC2.

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 AWSServiceRoleForECSrôle 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 AWSServiceRoleForECSservice. 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 AWSServiceRoleForECS lié au service. 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 en savoir plus, consultez Modification d'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 fonction 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 panneau de navigation, choisissez Rôles, puis choisissez le nom AWSServiceRoleForECS (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 AWSServiceRoleForECS rôle, 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 lié à AWSServiceRoleForECS un service

Vous devez supprimer tous les clusters Amazon ECS dans toutes les AWS régions avant de pouvoir supprimer le AWSServiceRoleForECS rôle.

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 Mise à jour d'un service à l'aide de la console et Suppression d'un service avec la console.
2. Forcez l'annulation de l'inscription de toutes les instances de conteneur de tous les clusters dans toutes les régions. Pour plus d'informations, consultez Annuler l'enregistrement d'une instance de conteneur sauvegardée par Amazon EC2.
3. Supprimez tous les clusters Amazon ECS dans toutes les régions. Pour plus d'informations, consultez Suppression d'un cluster avec la console.

Pour supprimer manuellement le rôle lié à un service à l'aide d'IAM

Utilisez la console IAM AWS CLI, le ou l' AWS API pour supprimer le rôle lié au AWSServiceRoleForECS service. 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.

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 Détails de l'autorisation.

La console Amazon ECS est alimentée par des autorisations IAM supplémentaires 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 IAM 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.

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-*"
            ]
      }
   ]
}
```

Si vous n'avez pas créé le rôle d'instance de conteneur Amazon ECS (`ecsInstanceRole`) et que vous créez un cluster qui utilise des instances Amazon EC2, 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 lors de l'utilisation de 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. Ceci est requis pour la gestion des groupes Amazon EC2 Auto Scaling lorsque vous utilisez la fonction de scalabilité automatique des clusters (Auto Scaling). 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 mandataires peuvent également répertorier les profils d'instance disponibles pour vos instances Amazon EC2.

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 IAM requises pour créer un fournisseur de capacité

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 fournisseurs de capacité Amazon ECS à l'aide de l' AWS Management Console et pour la gestion ultérieure de ces fournisseurs de capacité.

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/Infra-ECS-CapacityProvider-*"
            ]
        }
    ]
}
```

Autorisations IAM 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 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 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 IAM 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 (`escExternalInstanceRole`).

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 `escExternalInstanceRole` existant, vous devez disposer des autorisations `iam:GetRole` et `iam:PassRole`.

La politique suivante contient les autorisations requises et limite les actions au rôle `escExternalInstanceRole`.

```
{
  "Statement": [
      {
          "Effect": "Allow",
            "Action": [
```

```
                "iam:AttachRolePolicy",
                "iam:CreateRole",
                "iam:CreateInstanceProfile",
                "iam:AddRoleToInstanceProfile",
                "iam:ListInstanceProfilesForRole",
                "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/escExternalInstanceRole"
        },
        {
            "Effect": "Allow",
            "Action": ["iam:PassRole", "ssm:CreateActivation"],
            "Resource": "arn:aws:iam::*:role/escExternalInstanceRole"
        }
    ]
}
```

Autorisations IAM 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 IAM 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"
        }
    ]
```

```
}
```

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. Pour connaître les autorisations IAM dont votre application a besoin pour s'exécuter, veuillez consulter Rôle IAM de tâche.

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 à partir d'un référentiel privé Amazon ECR.
  - extrait une image de conteneur à partir 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 plus d'informations, consultez Utilisation du pilote du journal awslogs.
- Vos tâches sont hébergées sur des instances Amazon EC2 AWS Fargate ou sur des instances Amazon EC2 et...
  - utilisent l'authentification de registre privé. Pour plus d'informations, consultez Autorisations IAM requises pour l'authentification de registre privé.
  - utilise la surveillance du temps d'exécution.
  - la définition de tâche référence des données sensibles à l'aide de secrets Secrets Manager ou des paramètres AWS Systems Manager Parameter Store. Pour plus d'informations, consultez Autorisations IAM requises pour les secrets Amazon ECS.

> **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. Il peut être

nécessaire d'ajouter des politiques en ligne à votre rôle d'exécution de tâche pour les cas d'utilisation spéciaux décrits ci-dessous.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
    ]
}
```

Un rôle d'exécution de tâche Amazon ECS est automatiquement créé pour vous lors de la première exécution de la console Amazon ECS. Cependant, vous devez attacher manuellement la politique IAM gérée pour les tâches afin d'autoriser Amazon ECS à ajouter des autorisations pour les futures fonctions et améliorations au fur et à mesure qu'elles sont introduites. 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.

Création du rôle **`ecsTaskExecutionRole`** 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 un rôle IAM d'exécution de tâche (AWS Management Console)

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Rôles, puis Créer un rôle.

3. Dans la section Trusted entity type (Type d'entité de confiance), choisissez AWS service, Elastic Container Service.
4. Pour Use case (Cas d'utilisation), choisissez Elastic Container Service Task, puis Next (Suivant).
5. Dans la section Ajouter des autorisations, procédez comme suit :
   a. Recherchez AmazonECS TaskExecutionRolePolicy, puis sélectionnez la politique.
   b. Sous Set permissions boundary - optional (Définir la limite des autorisations - facultatif), choisissez Create role without a permissions boundary (Créer un rôle sans limite d'autorisations.
   c. Choisissez Suivant.
6. Sous Nom, révision et création, procédez comme suit :
   a. Pour Role name (Nom du rôle), tapez `ecsTaskExecutionRole`.
   b. Pour Add tags (optional) [Ajouter des étiquettes (facultatif)], spécifiez les étiquettes personnalisées à associer à la politique.
7. Sélectionnez Create role (Créer un rôle).

AWS CLI

Remplacez toutes les *`entrées utilisateur`* 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
```

Autorisations IAM requises pour l'authentification de registre privé

Le rôle d'exécution de tâche Amazon ECS est requis pour utiliser cette fonction. Cela permet à l'agent de conteneur d'extraire l'image de conteneur.

Pour fournir l'accès aux secrets que vous créez, ajoutez les autorisations suivantes en tant que politique en ligne au rôle d'exécution de tâche. Pour plus d'informations, consultez Ajout et suppression de politiques IAM.

- `secretsmanager:GetSecretValue`
- `kms:Decrypt` : requis uniquement si votre clé 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"
            ]
        }
    ]
}
```

Autorisations IAM requises pour les secrets Amazon ECS

Pour utiliser la fonction de secrets Amazon ECS, vous devez disposer du rôle d'exécution de tâche Amazon ECS et le référencer dans votre définition de tâche. Cela permet à l'agent du conteneur d'extraire les ressources nécessaires AWS Systems Manager ou les ressources de Secrets Manager. Pour plus d’informations, consultez Transmission de données sensibles à un conteneur.

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 lancement EC2, vous devez utiliser la variable de configuration de l'agent ECS `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` pour utiliser cette fonction. 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 plus d'informations, consultez 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"
      ]
    }
  ]
}
```

Autorisations IAM facultatives pour les tâches Fargate qui extraient des images Amazon ECR à partir des points de terminaison d'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.
- `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 :

> **Important**
>
> Les clés `aws:sourceVpc` ou les clés de `aws:sourceVpce` condition ne peuvent pas être appliquées à l'action `ecr:GetAuthorizationToken` API car l'appel d' GetAuthorizationToken API passe par l'interface Elastic Network appartenant à AWS Fargate plutôt que par l'interface réseau Elastic de la tâche.

```
{
```

```
    "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": {
                    "aws:sourceVpce": "vpce-xxxxxx",
                    "aws:sourceVpc": "vpc-xxxxx"
                }
            }
        }
    ]
}
```

Rôle IAM de tâche

Vos tâches Amazon ECS peuvent être associées à un rôle IAM. Les autorisations accordées dans le rôle IAM sont prises en charge par les conteneurs exécutés dans la tâche. 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.

Si vos applications conteneurisées doivent appeler des AWS API, elles doivent signer leurs demandes d' AWS API avec des AWS informations d'identification, et un rôle IAM de tâche fournit une stratégie de gestion des informations d'identification à utiliser par vos applications, de la même manière qu'un profil d'instance Amazon EC2 fournit des informations d'identification aux instances Amazon EC2. Au lieu de créer et distribuer vos informations d'identification AWS aux conteneurs ou d'utiliser le rôle de l'instance Amazon EC2, vous pouvez associer un rôle IAM à une définition de

tâche Amazon ECS ou une opération d'API `RunTask`. Vos conteneurs peuvent ensuite utiliser le AWS SDK ou AWS CLI envoyer des demandes d'API aux AWS services autorisés.

Les éléments suivants expliquent les avantages de l'utilisation de rôles IAM avec vos tâches.

- Isolement des informations d'identification : un conteneur ne peut récupérer des informations d'identification que pour le rôle IAM qui est défini dans la définition de tâche à laquelle il appartient ; un conteneur n'a jamais accès aux informations d'identification destinées à un autre conteneur appartenant à une autre tâche.
- Autorisation : les conteneurs non autorisés ne peuvent pas accéder aux informations d'identification du rôle IAM définies pour les autres tâches.
- Auditabilité : la journalisation des accès et des événements est disponible CloudTrail pour garantir un audit rétrospectif. Les informations d'identification des `taskArn` tâches ont un contexte associé à la session, de sorte que CloudTrail les journaux indiquent quelle tâche utilise quel rôle.

> **Note**
>
> Lorsque vous spécifiez un rôle IAM pour une tâche, les SDK AWS CLI ou les autres SDK présents dans les conteneurs de cette tâche utilisent exclusivement les AWS informations d'identification fournies par le rôle de tâche et n'héritent plus des autorisations IAM de l'instance Amazon EC2 ou externe sur laquelle ils s'exécutent.

Vous pouvez spécifier un rôle IAM de tâche à utiliser dans vos définitions de tâche. Vous pouvez aussi utiliser un remplacement `taskRoleArn` lorsque vous exécutez une tâche manuellement avec l'opération d'API `RunTask`. L'agent Amazon ECS reçoit un message de charge utile pour lancer la tâche avec des champs supplémentaires contenant les informations d'identification du rôle. L'agent Amazon ECS définit un ID unique d'informations d'identification de tâche en tant que jeton d'identification et met à jour son cache interne d'informations d'identification, de sorte que le jeton d'identification de la tâche pointe vers les informations d'identification du rôle qui sont reçues dans la charge utile. L'agent Amazon ECS remplit la variable d'environnement `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` dans l'objet `Env` (disponible avec la commande docker inspect ***`container_id`***) pour tous les conteneurs appartenant à cette tâche avec l'URI relatif suivant : `/`*`credential_provider_version`*`/credentials?id=`*`task_credential_id`*.

Depuis l'intérieur du conteneur, vous pouvez demander le point de terminaison d'informations d'identification en entrant la commande suivante :

```
curl 169.254.170.2$AWS_CONTAINER_CREDENTIALS_RELATIVE_URI
```

Sortie :

```
{
    "AccessKeyId": "ACCESS_KEY_ID",
    "Expiration": "EXPIRATION_DATE",
    "RoleArn": "TASK_ROLE_ARN",
    "SecretAccessKey": "SECRET_ACCESS_KEY",
    "Token": "SECURITY_TOKEN_STRING"
}
```

Si votre instance Amazon EC2 utilise au moins une version `1.11.0` de l'agent de conteneur et une version prise en charge des SDK, le client du SDK verra que la `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` variable est disponible et utilisera les informations d'identification fournies pour appeler les API. AWS CLI AWS Pour plus d'informations, consultez Utilisation des rôles IAM de tâche sur vos instances Amazon EC2 ou externes.

Chaque fois que le fournisseur d'informations d'identification est utilisé, la demande est enregistrée localement sur l'instance du conteneur hôte à l'adresse `/var/log/ecs/audit.log.`*`YYYY-MM-DD-HH`*. Pour plus d'informations, consultez Journal d'audit des informations d'identification des rôles IAM pour les tâches.

Considérations pour les tâches hébergées sur des instances Amazon EC2

Lorsque vous utilisez un rôle IAM avec vos tâches qui s'exécutent sur des instances Amazon EC2, les conteneurs ne sont pas empêchés d'accéder aux informations d'identification qui sont fournies au profil d'instance Amazon EC2 (par le biais du serveur de métadonnées d'instance Amazon EC2). 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`. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.

Les points suivants doivent également être pris en compte lors de l'utilisation d'un rôle IAM de tâche pour les tâches hébergées sur des instances Amazon EC2.

- Pour empêcher les conteneurs exécutés par des tâches utilisant le mode réseau `awsvpc` d'accéder aux informations d'identification fournies par le profil d'instance Amazon EC2, tout en accordant les autorisations fournies par le rôle de la tâche, définissez la variable de configuration

d'agent `ECS_AWSVPC_BLOCK_IMDS` sur `true` dans le fichier de configuration de l'agent et redémarrez l'agent. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

- Pour empêcher les conteneurs exécutés par des tâches utilisant le mode réseau `bridge` d'accéder aux informations d'identification fournies par le profil d'instance Amazon EC2, tout en accordant les autorisations fournies par le rôle de la tâche, exécutez la commande iptables suivante sur vos instances Amazon EC2. Cette commande n'a pas d'incidence sur les conteneurs dans les tâches utilisant le mode réseau `host` ou `awsvpc`. Pour plus d'informations, consultez 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 Amazon EC2. 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
  ```

Utilisation des rôles IAM de tâche sur vos instances Amazon EC2 ou externes

Pour utiliser les rôles IAM des tâches, vos instances Amazon EC2 ou externes doivent au moins disposer de la version `1.11.0` de l'agent de conteneur ; nous vous conseillons cependant 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 plus d'informations, consultez AMI optimisée pour Amazon ECS.

Si vous n'utilisez pas l'AMI optimisée pour Amazon ECS pour vos instances de conteneur, veillez à ajouter l'option `--net=host` à la commande docker run qui lance l'agent, ainsi que les variables de configuration de l'agent suivantes pour la configuration souhaitée (pour plus d'informations, consultez Configuration 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 des AMI non optimisées pour Amazon ECS). 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.

Création d'un rôle et d'une politique IAM pour vos tâches

Lors de la création d'une politique IAM que vos tâches utiliseront, la politique doit inclure les autorisations que les conteneurs de vos tâches doivent assumer. 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 de plus amples informations, veuillez consulter la rubrique Modifying a role trust policy (Modification d'une politique d'approbation de rôle) 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, veuillez consulter Points de terminaison de service dans le Guide de référence Référence générale d'Amazon Web Services (langue française non garantie).

Le rôle de tâche IAM doit posséder une politique de confiance qui spécifie le service `ecs-tasks.amazonaws.com`. L'autorisation `sts:AssumeRole` permet à vos tâches de prendre un rôle IAM différent de celui utilisé par votre instance Amazon EC2. De cette façon, votre tâche n'hérite pas du rôle associé à l'instance Amazon EC2. 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.

```
{
   "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"
            }
         }
      }
   ]
}
```

> **Important**
>
> Lorsque vous créez votre rôle IAM dans la tâche, il est recommandé d'utiliser les clés de `aws:SourceArn` condition `aws:SourceAccount` ou dans la relation de confiance ou dans la politique IAM associée au rôle afin d'étendre davantage les autorisations afin d'éviter tout problème de sécurité adjoint. 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.

Les procédures suivantes décrivent comment créer une politique pour récupérer des objets depuis Amazon S3 avec un exemple de politique. Remplacez toutes les *`entrées utilisateur`* 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 l'adresse https://console.aws.amazon.com/iam/.
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 politiques, 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/*"
         ],
         "Condition":{
            "ArnLike":{
            "aws:SourceArn":"arn:aws:ecs:region:123456789012:*"
            },
            "StringEquals":{
               "aws:SourceAccount":"123456789012"
            }
         }
      }
   ]
}
```

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 de plus amples 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 toutes les *entrées utilisateur* par vos propres valeurs.

1. Créez un fichier nommé `s3-policy.json` avec le contenu suivant.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
         ],
         "Resource":[
            "arn:aws:s3:::my-task-secrets-bucket/*"
         ],
         "Condition":{
            "ArnLike":{
            "aws:SourceArn":"arn:aws:ecs:region:123456789012:*"
            },
            "StringEquals":{
               "aws:SourceAccount":"123456789012"
            }
         }
      }
   ]
}
```

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 taskRolePolicy \
      --policy-document file://s3-policy.json
```

Les procédures suivantes décrivent comment créer un rôle IAM de tâche en joignant une politique IAM que vous créez.

AWS Management Console

Pour créer un rôle IAM pour vos tâches (AWS Management Console)

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Rôles, puis Créer un rôle.
3. Pour la section Select type of trusted entity (Sélectionner le type d'entité de confiance), choisissez AWS service.
4. Pour Use case (Cas d'utilisation), à l'aide du menu déroulant, sélectionnez Elastic Container Service, puis le cas d'utilisation Elastic Container Service Task, puis choisissez Next(Suivant).
5. Pour Ajouter des autorisations, recherchez et choisissez la politique que vous avez créée, puis choisissez Suivant.
6. A l'Étape 3 : Nommer, vérifier et créer, procédez comme suit :
   a. Dans le champ Role name (Nom de rôle), saisissez un nom pour votre rôle. Pour cet exemple, tapez `AmazonECSTaskS3BucketRole` comme nom du rôle.
   b. (Facultatif) Pour Description, spécifiez une description pour ce rôle IAM.
   c. Vérifiez l'entité de confiance et la politique d'autorisations associées au rôle.
   d. Pour Add tags (optional) (Ajouter des étiquettes (facultatif)), saisissez les étiquettes de métadonnées que vous souhaitez associer au rôle IAM, puis choisissez Create role (Créer un rôle).

AWS CLI

Remplacez toutes les *entrées utilisateur* par vos propres valeurs.

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.

```
{
   "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
```

Autorisations IAM requises pour 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.

```
{
   "Version": "2012-10-17",
   "Statement": [
       {
       "Effect": "Allow",
       "Action": [
            "ssmmessages:CreateControlChannel",
            "ssmmessages:CreateDataChannel",
            "ssmmessages:OpenControlChannel",
            "ssmmessages:OpenDataChannel"
       ],
      "Resource": "*"
      }
   ]
}
```

Spécification d'un rôle IAM pour vos tâches

Une fois que vous avez créé un rôle et lié une politique à ce rôle, vous pouvez exécuter les tâches qui assument le rôle. Plusieurs possibilités s'offrent à vous :

- Spécifier un rôle IAM pour vos tâches dans la définition de tâche. Vous pouvez créer une définition de tâche ou une révision d'une définition de tâche existante et spécifier le rôle que vous avez créé précédemment. Si vous utilisez la console pour créer votre définition de la tâche, sélectionnez le rôle IAM dans le champ Task Role (Rôle de la tâche). Pour plus d’informations, consultez Création d'une définition de tâche à l'aide de la console.

  Si vous utilisez les SDK AWS CLI ou, spécifiez l'Amazon Resource Name (ARN) de votre rôle de tâche à l'aide du `taskRoleArn` paramètre. Pour plus d'informations, consultez TaskDefinitionle manuel de référence des API Amazon ECS etParamètres de définition de tâche.

> **Note**
>
> Cette option est obligatoire si vous voulez utiliser des rôles de tâche IAM dans un service Amazon ECS service.

- Spécifiez un remplacement de rôle de tâche IAM lors de l'exécution d'une tâche. Vous pouvez spécifier un remplacement de rôle de tâche IAM lors de l'exécution d'une tâche. Si vous utilisez les SDK AWS CLI ou, spécifiez l'ARN de votre rôle de tâche à l'aide du `taskRoleArn` paramètre de l'objet `overrides` JSON. Pour plus d'informations sur le `overrides` paramètre, consultez RunTasket TaskOverridedans le manuel Amazon ECS API Reference. Pour plus d'informations sur le remplacement à l'aide de la console, consultezExécuter une application en tant que tâche Amazon ECS.

> **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.

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 EC2 de type 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.

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 Amazon ECS Service Connect, le CloudTrail journal le `roleSessionName`

seraECSServiceConnectForTLS. Vous pouvez utiliser ce nom pour rechercher des événements dans la CloudTrail console en filtrant par nom d'utilisateur.

Amazon ECS fournit les politiques gérées suivantes nommées AmazonECSInfrastructureRolePolicyForVolumes et AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity contenant les autorisations requises pour l'attachement de volume et le protocole TLS respectivement. Pour plus d'informations, consultez AWS politiques gérées pour Amazon Elastic Container Service.

AmazonECSInfrastructureRolePolicyForVolumes

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "CreateEBSManagedVolume",
   "Effect": "Allow",
   "Action": "ec2:CreateVolume",
   "Resource": "arn:aws:ec2:*:*:volume/*",
   "Condition": {
    "ArnLike": {
     "aws:RequestTag/AmazonECSCreated": "arn:aws:ecs:*:*:task/*"
    },
    "StringEquals": {
     "aws:RequestTag/AmazonECSManaged": "true"
    }
   }
  },
  {
   "Sid": "TagOnCreateVolume",
   "Effect": "Allow",
   "Action": "ec2:CreateTags",
   "Resource": "arn:aws:ec2:*:*:volume/*",
   "Condition": {
    "ArnLike": {
     "aws:RequestTag/AmazonECSCreated": "arn:aws:ecs:*:*:task/*"
    },
    "StringEquals": {
     "ec2:CreateAction": "CreateVolume",
     "aws:RequestTag/AmazonECSManaged": "true"
    }
   }
```

```
},
{
 "Sid": "DescribeVolumesForLifecycle",
 "Effect": "Allow",
 "Action": [
  "ec2:DescribeVolumes",
  "ec2:DescribeAvailabilityZones"
 ],
 "Resource": "*"
},
{
 "Sid": "ManageEBSVolumeLifecycle",
 "Effect": "Allow",
 "Action": [
  "ec2:AttachVolume",
  "ec2:DetachVolume"
 ],
 "Resource": "arn:aws:ec2:*:*:volume/*",
 "Condition": {
  "StringEquals": {
   "aws:ResourceTag/AmazonECSManaged": "true"
  }
 }
},
{
 "Sid": "ManageVolumeAttachmentsForEC2",
 "Effect": "Allow",
 "Action": [
  "ec2:AttachVolume",
  "ec2:DetachVolume"
 ],
 "Resource": "arn:aws:ec2:*:*:instance/*"
},
{
 "Sid": "DeleteEBSManagedVolume",
 "Effect": "Allow",
 "Action": "ec2:DeleteVolume",
 "Resource": "arn:aws:ec2:*:*:volume/*",
 "Condition": {
  "ArnLike": {
   "aws:ResourceTag/AmazonECSCreated": "arn:aws:ecs:*:*:task/*"
  },
  "StringEquals": {
   "aws:ResourceTag/AmazonECSManaged": "true"
```

```
    }
   }
  }
 ]
}
```

AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "CreateSecret",
            "Effect": "Allow",
            "Action": "secretsmanager:CreateSecret",
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*",
            "Condition": {
                "ArnLike": {
                    "aws:RequestTag/AmazonECSCreated": [
                        "arn:aws:ecs:*:*:service/*/*",
                        "arn:aws:ecs:*:*:task-set/*/*"
                    ]
                },
                "StringEquals": {
                    "aws:RequestTag/AmazonECSManaged": "true",
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
            }
        },
        {
            "Sid": "TagOnCreateSecret",
            "Effect": "Allow",
            "Action": "secretsmanager:TagResource",
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*",
            "Condition": {
                "ArnLike": {
                     "aws:RequestTag/AmazonECSCreated": [
                        "arn:aws:ecs:*:*:service/*/*",
                        "arn:aws:ecs:*:*:task-set/*/*"
                    ]
                },
                "StringEquals": {
                    "aws:RequestTag/AmazonECSManaged": "true",
```

```
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
            }
        },
        {
            "Sid": "RotateTLSCertificateSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:UpdateSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:DeleteSecret",
                "secretsmanager:RotateSecret",
                "secretsmanager:UpdateSecretVersionStage"
            ],
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:ResourceTag/aws:secretsmanager:owningService":
"ecs-sc",
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
            }
        },
        {
            "Sid": "ManagePrivateCertificateAuthority",
            "Effect": "Allow",
            "Action": [
                "acm-pca:GetCertificate",
                "acm-pca:GetCertificateAuthorityCertificate",
                "acm-pca:DescribeCertificateAuthority"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/AmazonECSManaged": "true"
                }
            }
        },
        {
            "Sid":
"ManagePrivateCertificateAuthorityForIssuingEndEntityCertificate",
            "Effect": "Allow",
```

```
            "Action": [
                "acm-pca:IssueCertificate"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/AmazonECSManaged": "true",
                    "acm-pca:TemplateArn":"arn:aws:acm-pca:::template/
EndEntityCertificate/V1"
                }
            }
        }
    ]
}
```

Création du rôle d'infrastructure ECS (**ecsInfrastructureRole**)

Remplacez toutes les *entrées utilisateur* 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 :

```
{
  "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 IAM 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. Selon votre cas d'utilisation, associez le AWS gestionnaire `AmazonECSInfrastructureRolePolicyForVolumes` ou la `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` politique au `ecsInfrastructureRole` rôle.

```
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 (https://console.aws.amazon.com/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.

Octroi à un utilisateur de l'autorisation de transmettre le rôle d'infrastructure ECS 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.

```
{
    "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.

Configuration supplémentaire des rôles IAM pour les tâches Windows

> **Important**
>
> Pour les conteneurs Windows sur Fargate qui utilisent des rôles de tâches, aucune autre action n'est nécessaire. Pour les conteneurs Windows sur EC2 qui utilisent des rôles de tâches, procédez comme suit.

Les rôles IAM pour les tâches avec des fonctions Windows exigent une configuration supplémentaire sur EC2, mais une grande partie de cette configuration est similaire à celle des rôles IAM pour les tâches sur les instances de conteneur Linux. Les exigences suivantes doivent être satisfaites pour configurer des rôles IAM pour des tâches sur les conteneurs Windows.

- 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 Rôles IAM pour le script d'amorçage du conteneur de tâches.
- Vous devez créer un rôle et une stratégie IAM pour vos tâches. Pour plus d’informations, consultez Création d'un rôle et d'une politique IAM pour vos tâches.

- Vous devez spécifier le rôle IAM que vous avez créé pour vos tâches lorsque vous enregistrez la définition de tâche, ou comme un remplacement lorsque vous exécutez la tâche. Pour plus d'informations, consultez Spécification d'un rôle IAM pour vos tâches.
- 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 plus d'informations, consultez Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge.
- 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.

Rôles IAM pour le script d'amorçage du conteneur de tâches

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 Amazon EC2 et les instances externes, exécutent l'agent de conteneur Amazon ECS et utilisent un rôle IAM pour indiquer au service 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 plus d'informations, consultez Rôle IAM 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 à l'agent de conteneur Amazon ECS et au daemon Docker pour appeler des AWS API en votre nom. Pour de plus amples informations sur la stratégie gérée, veuillez consulter Rôle Amazon EC2 EC2 ContainerServicefor.

Amazon ECS prend en charge le lancement d'instances de conteneur avec une augmentation de la densité ENI et utilisant des types d'instances Amazon EC2 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, consultezAccès aux fonctionnalités d'Amazon ECS via les paramètres du compte.

Création du rôle d'instance de conteneur (**`ecsInstanceRole`**)

> **Important**
>
> Si vous enregistrez des instances externes dans votre cluster, veuillez consulter Rôle IAM 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 IAM **`ecsInstanceRole`** pour vos instances de conteneur

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation, sélectionnez Rôles, puis Créer un rôle.
3. Choisissez le type AWS de rôle de service, puis sous Cas d'utilisation pour d'autres AWS services, choisissez Elastic Container Service.
4. Choisissez le cas d'utilisation EC2 Role for Elastic Container Service (Rôle EC2 pour Elastic Container Service), puis sélectionnez Next: Permissions (Suivant : Autorisations).
5. Dans la section Politiques d'autorisation, vérifiez que la politique Amazon EC2 ContainerServicefor EC2Role est sélectionnée, puis choisissez Next.

   > **Important**
   >
   > La politique gérée Amazon EC2 ContainerServicefor EC2Role doit être attachée au rôle IAM de l'instance de conteneur, sinon vous recevrez un message d'erreur lors de l'utilisation de pour créer des clusters. AWS Management Console

6. Dans Nom du rôle, entrez ecsInstanceRoleet vous pouvez éventuellement saisir une description.
7. Pour Add tags (optional) (Ajouter des étiquettes [facultatif]), entrez les étiquettes personnalisées à associer à la stratégie, puis choisissez Next: Review (Suivant : Vérifier).
8. Passez en revue les informations de votre rôle, puis choisissez Create role (Créer un rôle) pour terminer.

AWS CLI

Remplacez toutes les *`entrées utilisateur`* par vos propres valeurs.

1. Créez un fichier nommé `instance-role-trust-policy.json` avec le contenu suivant.

```
{
  "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
```

Ajout de l'accès en lecture seule Amazon S3 à votre rôle (**ecsInstanceRole**) d'instance de conteneur

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 utilisateur Amazon EC2 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 d'instance de conteneur dans Simple Storage Service (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 Modification d'une politique d'autorisations de rôle (console) dans le Guide de AWS Identity and Access Management l'utilisateur.

Autorisations requises pour surveiller les instances de conteneur

Avant que vos instances de conteneur puissent envoyer des données de journal à CloudWatch Logs, vous devez créer une stratégie IAM pour permettre à vos instances de conteneur d'utiliser les API CloudWatch Logs, puis vous devez associer cette politique 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 l'adresse https://console.aws.amazon.com/iam/.
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 politiques, 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 de plus amples 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 Modification d'une politique d'autorisations de rôle (console) 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.

```
{
    "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. Remplacez *cwlogspolicy* par le nom de la politique que vous avez créée.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`cwlogsppolicy`].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 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 AWS les API. 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.

Création du rôle ECS Anywhere (**`ecsAnywhereRole`**)

Remplacez toutes les *`entrées utilisateur`* par vos propres informations.

1. Créez un fichier local nommé `ssm-trust-policy.json` avec la politique de confiance suivante.

```
{
  "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 personnalisée de la console IAM (https://console.aws.amazon.com/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.

Autorisations IAM conditionnelles

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 plus d’informations, consultez 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. Cependant, si le rôle IAM `ECSAnywhereRole` associé à votre instance externe inclut également les autorisations nécessaires pour extraire des images d'Amazon ECR, alors votre rôle d'exécution de tâche n'a pas besoin de les inclure.

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 pour le déploiement bleu/vert.

Deux politiques gérées sont fournies. La `AWSCodeDeployRoleForECS` politique, illustrée ci-dessous, CodeDeploy autorise la mise à jour de toute ressource à l'aide de l'action associée. La `AWSCodeDeployRoleForECSLimited` politique, présentée ci-dessous, donne des autorisations CodeDeploy plus limitées.

AWSCodeDeployRoleForECS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ecs:DescribeServices",
                "ecs:CreateTaskSet",
                "ecs:UpdateServicePrimaryTaskSet",
                "ecs:DeleteTaskSet",
                "elasticloadbalancing:DescribeTargetGroups",
                "elasticloadbalancing:DescribeListeners",
                "elasticloadbalancing:ModifyListener",
                "elasticloadbalancing:DescribeRules",
                "elasticloadbalancing:ModifyRule",
                "lambda:InvokeFunction",
                "cloudwatch:DescribeAlarms",
```

```
                "sns:Publish",
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": ["iam:PassRole"],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": ["ecs-tasks.amazonaws.com"]
                }
            }
        }
    ]
}
```

AWSCodeDeployRoleForECSLimited

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ecs:DescribeServices",
                "ecs:CreateTaskSet",
                "ecs:UpdateServicePrimaryTaskSet",
                "ecs:DeleteTaskSet",
                "cloudwatch:DescribeAlarms"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": ["sns:Publish"],
            "Resource": "arn:aws:sns:*:*:CodeDeployTopic_*",
            "Effect": "Allow"
        },
        {
            "Action": [
```

```
                "elasticloadbalancing:DescribeTargetGroups",
                "elasticloadbalancing:DescribeListeners",
                "elasticloadbalancing:ModifyListener",
                "elasticloadbalancing:DescribeRules",
                "elasticloadbalancing:ModifyRule"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": ["lambda:InvokeFunction"],
            "Resource": "arn:aws:lambda:*:*:function:CodeDeployHook_*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"s3:ExistingObjectTag/UseWithCodeDeploy": "true"}
            },
            "Effect": "Allow"
        },
        {
            "Action": ["iam:PassRole"],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:iam::*:role/ecsTaskExecutionRole",
                "arn:aws:iam::*:role/ECSTaskExecution*"
            ],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": ["ecs-tasks.amazonaws.com"]
                }
            }
        }
    ]
}
```

Création du CodeDeploy **`AWSCodeDeployRoleForECS`** rôle

Vous pouvez utiliser les procédures suivantes pour créer un CodeDeploy rôle pour Amazon ECS

AWS Management Console

Pour créer un rôle IAM pour CodeDeploy

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, choisissez Rôles, puis Créer un rôle.
3. Pour la section Select type of trusted entity (Sélectionner le type d'entité de confiance), choisissez AWS service.
4. Pour Choisir le service qui utilisera ce rôle, choisissez CodeDeploy.
5. Pour Sélectionnez votre cas d'utilisation, choisissez CodeDeploy - ECS, Next.
6. Dans la section Ajouter des autorisations, procédez comme suit

   a. Assurez-vous que la AWSCodeDeployRoleForECSpolitique est sélectionnée.

   b. Sous Set permissions boundary - optional (Définir la limite des autorisations - facultatif), choisissez Create role without a permissions boundary (Créer un rôle sans limite d'autorisations.

   c. Choisissez Suivant.
7. Sous Nom, révision et création, procédez comme suit :

   a. Pour Role name (Nom du rôle), entrez `ecsCodeDeployRole` et, si vous le souhaitez, vous pouvez entrer une description facultative.

   b. Pour Add tags (optional) [Ajouter des étiquettes (facultatif)], entrez les étiquettes personnalisées à associer à la politique.
8. Sélectionnez Créer un rôle.

AWS CLI

Remplacez toutes les *`entrées utilisateur`* 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.

```
{
```

```
    "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
```

Ajout d’autorisations pour les déploiements bleu/vert

Si les tâches de votre service Amazon ECS utilisant le type de déploiement bleu/vert nécessitent l'utilisation du rôle d'exécution de tâches ou un remplacement de rôle de tâche, vous devez ajouter l'`iam:PassRole`autorisation pour chaque rôle d'exécution de tâche ou de remplacement de rôle de tâche au rôle CodeDeploy IAM en tant que politique. Pour plus d’informations, consultez Rôle IAM d'exécution de tâche Amazon ECS et Rôle IAM de tâche.

Utilisez la procédure suivante pour créer la politique

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 l'adresse https://console.aws.amazon.com/iam/.
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 politiques, 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 de plus amples 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, consultez la section Modification d'une politique d'autorisations de rôle (console) dans le Guide de AWS Identity and Access Management l'utilisateur.

AWS CLI

Remplacez toutes les *entrées utilisateur* par vos propres informations.

1. Créez un fichier nommé `blue-green-iam-passrole.json` avec le contenu suivant.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsTaskExecutionRole_or_TaskRole_name>"]
        }
    ]
}
```

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.

```
{
    "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 tâche, 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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>: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 plus d'informations, consultez Exécutez des tâches Amazon ECS selon des plannings à l'aide d'Amazon EventBridge Scheduler.

Création du rôle Amazon ECS EventBridge (**`ecsEventsRole`**)

Remplacez toutes les *`entrées utilisateur`* 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 :

   ```
   {
     "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-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 personnalisée de la console IAM (https://console.aws.amazon.com/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 l'adresse https://console.aws.amazon.com/iam/.
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 politiques, 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 de plus amples 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, consultez la section Modification d'une politique d'autorisations de rôle (console) dans le Guide de AWS Identity and Access Management l'utilisateur.

AWS CLI

Remplacez toutes les *entrées utilisateur* par vos propres informations.

1. Créez un fichier nommé `ev-iam-passrole.json` avec le contenu suivant.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>: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 IAM requises pour Service Auto Scaling

Service Auto Scaling est rendu possible par une combinaison des API Amazon ECS et Application Auto Scaling. 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 Exemple de création de service et Exemple de mise à jour du service 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 plus d'informations, consultez 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 des ressources Amazon ECS.

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": {
              "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`.

Contrôler 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 valeur multiples, consultez Création d'une condition qui teste plusieurs valeurs de clés dans le IAM Guide de l'utilisateur.

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 .

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

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 en fonction des balises

La politique suivante permet aux utilisateurs de supprimer des clusters lorsque la balise comporte une paire clé/valeur « Objectif/Test ».

```
{
  "Version": "2012-10-17",
  "Statement": [
       {
      "Action": [
        "ecs:DeleteCluster"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:ecs:region:account-id: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

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 stratégies 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 (ACL), vous pouvez utiliser ces politiques pour donner l'accès à vos ressources.

Pour en savoir plus, 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 découvrir quelle est la différence entre l'utilisation des rôles et l'utilisation des politiques basées sur les ressources pour l'accès intercompte, consultez Différence entre les rôles IAM et les politiques basées sur les ressources dans le Guide de l'utilisateur IAM.

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 se trouvent dans un état particulier ; l'état doit avoir changé et être maintenu pendant un certain nombre de périodes. Pour plus d’informations, consultez 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 étendre et étendre les tâches de votre service en CloudWatch fonction de mesures telles que l'utilisation du processeur et de la mémoire. Pour plus d’informations, consultez Faites évoluer automatiquement votre service Amazon ECS.

Pour les clusters dont les tâches ou les services utilisent le type de lancement EC2, 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 employant le type de lancement EC2.

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 Qu'est-ce qu'Amazon CloudWatch Events ? dans le guide de l'utilisateur d'Amazon CloudWatch Events.

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 qui a été 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 plus d'informations, consultez 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 AWS 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 plus d'informations, consultez 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 :

- Guides de démarrage rapide sur la sécurité et la conformité : ces guides de déploiement abordent les considérations architecturales et indiquent les étapes à suivre pour déployer des environnements de base axés sur AWS la sécurité et la conformité.
- Architecture axée sur la sécurité et la conformité HIPAA sur Amazon Web Services : ce livre blanc décrit comment les entreprises peuvent créer des applications AWS conformes à la loi HIPAA.

  > **Note**
  >
  > Tous ne Services AWS sont pas éligibles à la loi HIPAA. Pour plus d'informations, consultez le HIPAA Eligible Services Reference.

- 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.
- 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.

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, veuillez consulter Federal Information Processing Standard (FIPS) 140-2.

Considérations

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) .
- 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 plus d'informations, consultez 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 *my-cluster* par le nom de votre cluster.

   Une valeur renvoyée de « 1 » indique que vous utilisez FIPS.

   ```
   aws ecs execute-command --cluster cluster-name \
       --interactive \
       --command "cat /proc/sys/crypto/fips_enabled"
   ```

Utilisation à CloudTrail des fins d'audit

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 plus d'informations, consultez 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 des politiques Amazon ECS pour contrôler l'accès à partir de points de terminaison Amazon Virtual Private Cloud (Amazon VPC) ou de VPC spécifiques. En fait, cela isole l'accès réseau à une ressource Amazon ECS donnée uniquement du VPC spécifique au sein AWS du réseau. Pour plus d'informations, consultez 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 d'interface sont AWS PrivateLink alimentés par une technologie qui vous permet d'accéder en privé aux API Amazon ECS en utilisant 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 sur AWS PrivateLink les points de terminaison VPC, consultez la section Points de terminaison VPC dans le guide de l'utilisateur Amazon VPC.

Considérations relatives aux points de terminaison d'un VPC Amazon ECS

Éléments à prendre en compte pour les points de terminaison d'un VPC Amazon ECS pour le type de lancement 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 d’un VPC d'interface (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.

    Si votre VPC ne possède pas de passerelle Internet, vous devez créer le point de terminaison de passerelle pour Amazon S3. Pour de plus amples informations, consultez Créer le point de terminaison de passerelle Amazon S3 dans le Guide de l'utilisateur Amazon Elastic Container Registry. Les points de terminaison d'interface pour Amazon S3 ne peuvent pas être utilisés avec Amazon ECR.

    > **Important**
    >
    > Si vous configurez Amazon ECR pour utiliser un point de terminaison d’un 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 d’un VPC spécifique. Pour plus d’informations, consultez Autorisations IAM facultatives pour les tâches Fargate qui extraient des images Amazon ECR à partir des points de terminaison d'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).

Éléments à prendre en compte pour les points de terminaison d'un 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 qui utilisent le type de lancement EC2 nécessitent que les instances de conteneur sur lesquelles elles sont lancées exécutent la version `1.25.1` ou ultérieure de l'agent de conteneur Amazon ECS. Pour plus d'informations, consultez Gestion des instances de conteneur Linux.
- 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.

Création de points de terminaison de VPC pour Amazon ECS

Pour créer les points de terminaison d'un VPC pour le service Amazon ECS service, utilisez la procédure Créer un point de terminaison d'interface que vous trouverez dans le Guide de l'utilisateur Amazon VPC. 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.

- `com.amazonaws.`*`region`*`.ecs-agent`

- `com.amazonaws.region.ecs-telemetry`
- `com.amazonaws.region.ecs`

> **Note**
>
> *region* représente l'identifiant de région d'une région AWS prise en charge par Amazon ECS, telle que `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 lancement EC2, une fois que vous avez créé les points de terminaison d'un VPC, chaque instance de conteneur a besoin de 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":[
    {
      "Effect": "Allow",
      "Action": [
        "ecs:CreateCluster",
        "ecs:ListClusters"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Bonnes pratiques de sécurité

Ce guide fournit des recommandations de sécurité et de conformité pour protéger vos informations, systèmes et autres ressources qui dépendent d'Amazon ECS. Il présente également des évaluations des risques et des stratégies d'atténuation que vous pouvez utiliser pour mieux maîtriser les contrôles de sécurité conçus pour les clusters Amazon ECS et les charges de travail qu'ils prennent en charge. Chaque rubrique de ce guide commence par un bref aperçu, suivi d'une liste de recommandations et de bonnes pratiques que vous pouvez utiliser pour sécuriser vos clusters Amazon ECS.

Rubriques

- AWS Identity and Access Management
- Utilisation de rôles IAM avec les tâches Amazon ECS
- Sécurité du réseau
- Gestion des secrets
- Utilisation d'informations d'identification de sécurité temporaires avec des opérations d'API
- Conformité et sécurité
- Journalisation et surveillance
- AWS Fargate sécurité
- Considérations relatives à la sécurité d'instance de conteneur EC2
- Sécurité des tâches et des conteneurs
- Sécurité de l'exécution
- Bonnes pratiques en matière d'AMI
- AWS Les partenaires

AWS Identity and Access Management

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, veuillez consulter Présentation de la gestion des accès : autorisations et politiques.

Gestion de l'accès à Amazon ECS

Vous pouvez contrôler l'accès à Amazon ECS en créant et en appliquant des politiques IAM. Ces politiques sont composées d'un ensemble d'actions qui s'appliquent à un ensemble spécifique de ressources. L'action d'une politique définit la liste des opérations (telles que les API Amazon ECS) autorisées ou refusées, tandis que la ressource contrôle les objets Amazon ECS auxquels l'action s'applique. Des conditions peuvent être ajoutées à une politique pour en réduire le champ d'application. Par exemple, une politique peut être écrite pour autoriser uniquement l'exécution d'une action sur des tâches comportant un ensemble de balises particulier. Pour plus d'informations,

veuillez consulter Fonctionnement d'Amazon ECS avec IAM dans le Guide du développeur Amazon Elastic Container Service (langue française non garantie).

Recommandations

Nous vous recommandons d'effectuer les opérations suivantes lorsque vous configurez vos rôles et politiques IAM.

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.

> **Note**
>
> Lorsque vous référencez un ARN dans une politique, utilisez le nouveau format d'ARN plus long. Pour plus d'informations, veuillez consulter Amazon Resource Names (ARN) et ID dans le Guide du développeur Amazon Elastic Container Service (langue française non garantie).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:StopTask"
      ],
      "Condition": {
        "ArnEquals": {
          "ecs:cluster": "arn:aws:ecs:region:account_id:cluster/cluster_name"
        }
      },
      "Resource": [
```

```
        "arn:aws:ecs:region:account_id:task-definition/task_family:*"
      ]
    }
  ]
}
```

Laissez la ressource du cluster servir 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.

Isolez les utilisateurs finaux de l'API Amazon ECS en créant des pipelines automatisés

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.

```
{
  "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 principal IAM démarre et arrête toutes les tâches avec une clé de balise `Department` et une valeur de balise de `Accounting`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
            "Resource": "arn:aws:ecs:*",
            "Condition": {
                "StringEquals": {"ecs:ResourceTag/Department": "Accounting"}
            }
        }
    ]
}
```

Auditez régulièrement l'accès aux API Amazon ECS

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 aux API Amazon ECS 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, veuillez consulter Consignes pour les audits de sécurité Amazon ECS dans le Référence générale d'Amazon Web Services.

Utilisation de rôles IAM avec les tâches Amazon ECS

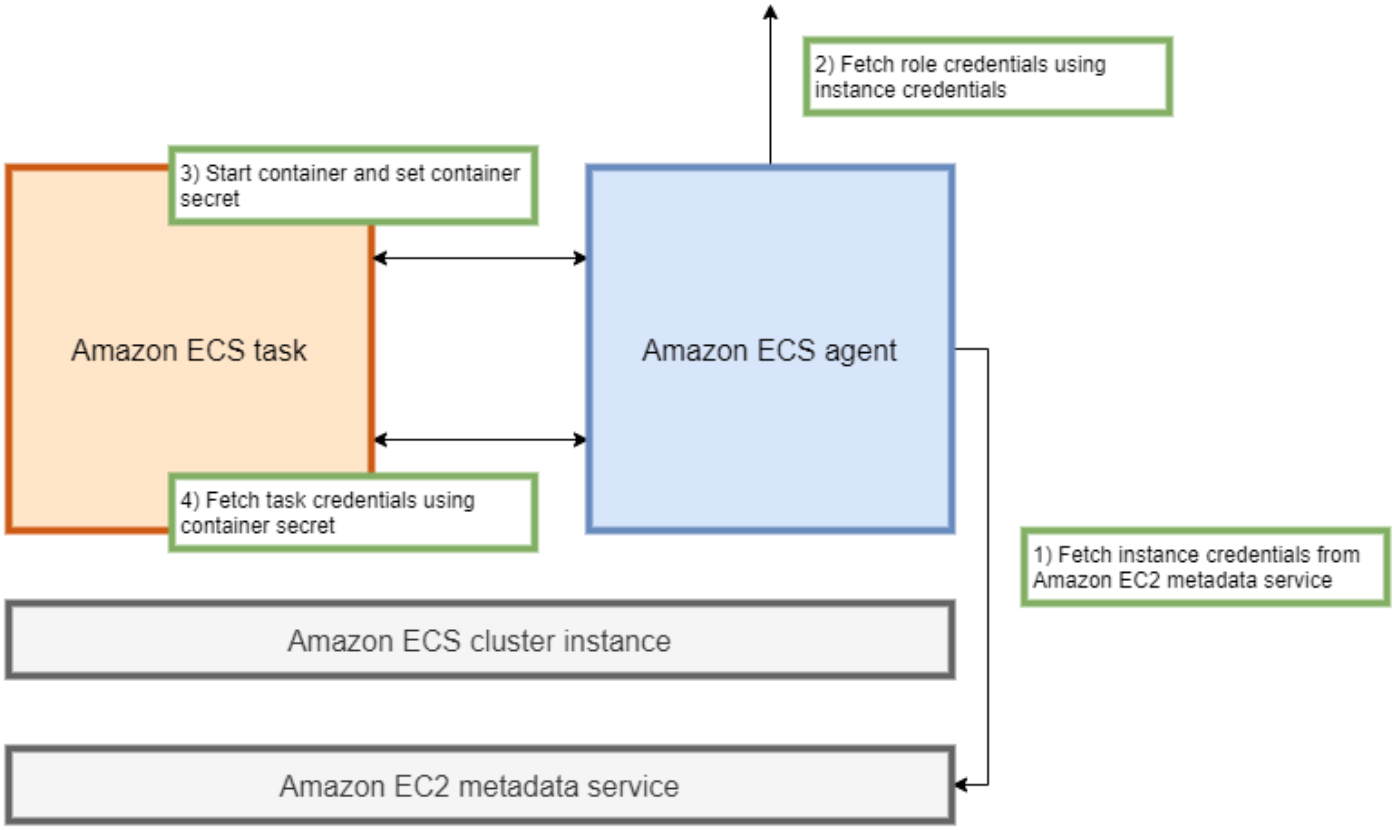
Nous vous recommandons d'attribuer un rôle IAM à une tâche. Son rôle peut être différencié du rôle de l'instance Amazon EC2 sur laquelle il s'exécute. L'attribution d'un rôle à chaque tâche est

conforme au principe de l'accès au moindre privilège et permet un contrôle plus précis des actions et des ressources.

Lorsque vous attribuez des rôles IAM à une tâche, vous devez utiliser la stratégie d'approbation suivante afin que chacune de vos tâches puisse endosser un rôle IAM différent de celui utilisé par votre instance EC2. De cette façon, votre tâche n'hérite pas du rôle de votre instance EC2.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

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 des AWS SDK récupèrent automatiquement ces informations d'identification à partir de la variable d'`AWS_CONTAINER_CREDENTIALS_RELATIVE_URI`environnement lors des appels d' AWS API.

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 d'instance de conteneur Amazon EC2

L'agent de conteneur Amazon ECS est un conteneur qui s'exécute sur chaque instance Amazon EC2 dans un cluster Amazon ECS. Il est initialisé en dehors d'Amazon ECS à l'aide de la commande `init` disponible sur le système d'exploitation. Par conséquent, aucune autorisation ne peut lui être accordée par le biais d'un rôle de tâche. Les autorisations doivent plutôt être attribuées aux instances Amazon EC2 sur lesquelles les agents s'exécutent. La liste des actions figurant dans l'exemple de politique `AmazonEC2ContainerServiceforEC2Role` doit être accordée à `ecsInstanceRole`. Si vous ne le faites pas, vos instances ne peuvent pas rejoindre le cluster.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeTags",
                "ecs:CreateCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:DiscoverPollEndpoint",
                "ecs:Poll",
                "ecs:RegisterContainerInstance",
                "ecs:StartTelemetrySession",
                "ecs:UpdateContainerInstancesState",
                "ecs:Submit*",
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
    ]
}
```

Dans cette politique, les actions `ecr` et `logs` API permettent aux conteneurs qui s'exécutent sur vos instances d'extraire des images d'Amazon ECR et d'écrire des journaux sur Amazon CloudWatch. Les actions `ecs` permettent à l'agent d'enregistrer et d'annuler l'enregistrement des instances et de communiquer avec le plan de contrôle Amazon ECS. Parmi celles-ci, l'action `ecs:CreateCluster` est facultative.

Rôles liés à un service

Vous pouvez utiliser le rôle lié à un service pour Amazon ECS pour octroyer au service Amazon ECS l'autorisation d'appeler d'autres API de service en votre 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

Nous vous recommandons d'effectuer les opérations suivantes lorsque vous configurez les rôles et politiques IAM de votre tâche.

Bloquez l'accès aux métadonnées Amazon EC2

Lorsque vous exécutez vos tâches sur des instances Amazon EC2, nous vous recommandons vivement de bloquer l'accès aux métadonnées Amazon EC2 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 métadonnées Amazon EC2, 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 192.0.2.0/32 --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 variable `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` sur `false` dans le fichier `ecs-agent config` pour empêcher les conteneurs exécutés sur le réseau `host` d'accéder aux métadonnées Amazon EC2.

Utilisez le mode réseau **`awsvpc`**

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 réseau `awsvpc` 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.

Utilisez IAM Access Advisor 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 résultats produits par IAM Access Advisor, 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, veuillez consulter IAM Access Advisor.

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 ayant contribué aux API.

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 endossent un rôle sont exécutées sur des instances de conteneur Amazon EC2, une demande est journalisée par l'agent de conteneur Amazon ECS dans le journal d'audit de l'agent situé à une adresse au format `/var/log/ecs/audit.log.YYYY-`

`MM-DD-HH`. Pour plus d'informations, veuillez consulter Journal des rôles IAM pour les tâches et Journalisation des événements Insights pour les journaux de suivi.

Sécurité du réseau

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.

- Avec un maillage de services (TLS) :

  Avec AWS App Mesh, vous pouvez configurer les connexions TLS entre les proxys Envoy déployés avec des points de terminaison maillés. Les nœuds virtuels et les passerelles virtuelles en sont deux exemples. Les certificats TLS peuvent provenir de AWS Certificate Manager (ACM). Ils peuvent également provenir de votre propre autorité de certification privée.

  - Activation du protocole TLS (Transport Layer Security)
  - Activez le chiffrement du trafic entre les services à AWS App Mesh l'aide de certificats ACM ou de certificats fournis par le client
  - Démonstration TLS avec ACM
  - Démonstration TLS avec des fichiers
  - Envoy
- 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

- nd-to-end Chiffrement électronique avec 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.

AWS PrivateLink

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 stratégies vous permettent d'affiner l'accès aux API d'un service. 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 d'agent de conteneur Amazon ECS

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 utilisées pour bloquer l'accès d'une tâche aux métadonnées Amazon EC2. `HTTP_PROXY` est utilisée 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

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

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 accessible au public, utilisez le protocole TLS/SSL pour chiffrer le trafic entre le navigateur du client et l'équilibreur de charge, puis chiffrez-le à nouveau vers le backend si cela est justifié.

Utilisez le mode réseau **`awsvpc`** et les groupes de sécurité lorsque vous devez contrôler le trafic entre les tâches ou entre les tâches et les autres ressources du réseau

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 repose sur 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 dans des Amazon VPC distincts lorsque le trafic réseau doit être strictement isolé

Vous devez créer des clusters dans des Amazon VPC distincts 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 des Amazon VPC distincts et exposez les services de manière sélective à d'autres Amazon VPC à l'aide des points de terminaison Amazon VPC. Pour plus d'informations, veuillez consulter Points de terminaison Amazon VPC.

Configurez les points de terminaison AWS PrivateLink lorsque cela est justifié

Vous devez configurer les AWS PrivateLink points de terminaison lorsque cela est justifié. Si votre politique de sécurité vous empêche de connecter une passerelle Internet (IGW) à vos Amazon VPC,

configurez les points de AWS PrivateLink terminaison pour Amazon ECS et d'autres services tels qu'Amazon ECR et Amazon. AWS Secrets Manager CloudWatch

Utilisez les journaux de flux Amazon VPC pour analyser le trafic à destination et en provenance de tâches de longue durée

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.

Gestion des secrets

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 à partir AWS Secrets Manager d'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).

Recommandations

Nous vous recommandons d'effectuer les opérations suivantes lorsque vous configurez la gestion des secrets.

Utilisez AWS Secrets Manager Amazon EC2 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 informations secrètes dans AWS Secrets Manager ou sous forme de paramètre chiffré dans Amazon EC2 Systems Manager Parameter Store. Ces services sont similaires car il s'agit tous deux de magasins de valeurs clés gérés utilisés AWS KMS pour chiffrer des données sensibles. AWS Secrets Manager, cependant, inclut également la possibilité de faire pivoter automatiquement les secrets, de générer des secrets aléatoires et de partager des secrets entre les AWS comptes. Si vous considérez ces fonctionnalités comme importantes, utilisez AWS Secrets Manager . Sinon, utilisez des paramètres chiffrés.

> **Note**
>
> Les tâches qui font référence à un secret provenant AWS Secrets Manager d'Amazon EC2 Systems Manager Parameter Store nécessitent un rôle d'exécution de tâches avec une politique qui accorde à Amazon ECS l'accès au secret souhaité et, le cas échéant, à AWS KMS la clé utilisée pour chiffrer et déchiffrer ce secret.

> **Important**
>
> Les secrets référencés dans les tâches ne font pas l'objet d'une rotation automatique. 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 :

- AWS Secrets Manager : injection de données en tant que variables d'environnement
- Amazon EC2 Systems Manager Parameter Store : injection de données en tant que variables d'environnement

Récupérez des données à partir d'un compartiment chiffré Amazon S3

Étant donné que la valeur des variables d'environnement peut être divulguée par inadvertance dans les journaux et être révélée lors de l'exécution de `docker inspect`, vous devez stocker les secrets dans un compartiment chiffré Amazon S3 et utiliser des rôles de tâche pour restreindre l'accès à ces secrets. 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 de conteneur sidecar, consultez le aws-secret-sidecar-injectorprojet.

> **Note**
>
> Sur Amazon EC2, le volume sur lequel le secret est écrit peut être chiffré à l'aide d'une clé AWS KMS gérée par le client. Activé AWS Fargate, le stockage en volume est automatiquement chiffré à l'aide d'une clé gérée par le service.

Ressources supplémentaires

- Transmission de secrets à des conteneurs dans une tâche Amazon ECS
- Chamber, encapsuleur permettant de stocker des secrets dans Amazon EC2 Systems Manager Parameter Store

Utilisation d'informations d'identification de sécurité temporaires avec des opérations d'API

Si vous envoyez des demandes d'API HTTPS directes à AWS, vous pouvez signer ces demandes avec les informations d'identification de sécurité temporaires que vous obtenez auprès du AWS Security Token Service. Pour plus d'informations, consultez la section Signature des demandes d' AWS API dans le Références générales AWS.

Conformité et sécurité

Votre responsabilité de conformité lors de l'utilisation d'Amazon ECS est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise, ainsi que par la législation et la réglementation en vigueur.

AWS fournit les ressources suivantes pour faciliter la mise en conformité :

- Guides de démarrage rapide sur la sécurité et la conformité : Ces guides de déploiement abordent les considérations architecturales et indiquent les étapes à suivre pour déployer des environnements de base axés sur la sécurité et la conformité sur. AWS
- Livre blanc sur l'architecture pour la sécurité et la conformité HIPAA : Ce livre blanc décrit comment les entreprises peuvent créer des applications conformes à la loi HIPAA. AWS
- Services AWS concernés par le programme de conformité : cette liste contient les services AWS concernés par les programmes de conformité spécifiques. Pour plus d'informations, veuillez consulter Programmes de conformitéAWS.

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
- Architecture pour le champ d'application et la segmentation de la norme PCI DSS sur AWS

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 ou Weave Net) pour garantir le chiffrement complet des PHI transférées entre les conteneurs ou pour fournir une couche de chiffrement redondante. La journalisation complète doit également être activée et tous les journaux des conteneurs doivent être dirigés vers Amazon CloudWatch. 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.

AWS Security Hub

AWS Security Hub À utiliser pour surveiller votre utilisation d'Amazon ECS en ce qui concerne les meilleures pratiques de sécurité. Security Hub utilise des contrôles pour évaluer les configurations des ressources et les normes de sécurité afin de vous aider à respecter divers cadres de conformité. Pour plus d'informations sur l'utilisation de Security Hub pour évaluer les ressources Amazon ECS, veuillez consulter Contrôles d'Amazon ECS dans le Guide de l'utilisateur AWS Security Hub (langue française non garantie).

Surveillance du temps GuardDuty d'exécution d'Amazon

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 et EC2 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 instances Amazon EC2 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

Vous devez impliquer rapidement les responsables des programmes de conformité au sein de votre entreprise et utiliser le modèle de responsabilité partagée AWS pour identifier les responsables du contrôle de conformité afin de réussir avec ces programmes.

Journalisation et surveillance

La journalisation et la surveillance sont des aspects importants du maintien de la fiabilité, de la disponibilité et des performances d'Amazon ECS et de vos AWS solutions. AWS fournit plusieurs outils pour surveiller vos ressources Amazon ECS et répondre aux incidents potentiels :

- CloudWatchAlarmes Amazon
- Amazon CloudWatch Logs
- CloudWatch Événements Amazon
- AWS CloudTrail Journaux

Vous pouvez configurer les conteneurs de vos tâches pour envoyer des informations de journal à Amazon CloudWatch Logs. Si vous utilisez le type de AWS Fargate lancement pour vos tâches, vous pouvez consulter les journaux de vos conteneurs. Si vous utilisez le type de lancement Amazon EC2, vous pouvez afficher différents journaux de vos conteneurs dans un emplacement pratique. Cela évite également que vos journaux de conteneur n'occupent de l'espace disque sur vos instances de conteneur.

Pour plus d'informations sur Amazon CloudWatch Logs, consultez la section Monitor Logs from Amazon EC2 Instances dans le guide de CloudWatch l'utilisateur Amazon. Pour obtenir des instructions sur l'envoi de journaux de conteneurs depuis vos tâches vers Amazon CloudWatch Logs, consultez la section Utilisation du pilote de `awslogs` journal.

Journalisation des conteneurs avec Fluent Bit

AWS fournit une Fluent Bit image avec des plugins pour Amazon CloudWatch Logs et Amazon Data Firehose. Cette image permet d'acheminer les journaux vers les destinations Amazon CloudWatch et Amazon Data Firehose (notamment Amazon S3, Amazon OpenSearch Service et Amazon Redshift). 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 Amazon CloudWatch Logs pour Fluent Bit et Amazon Data Firehose pour. Fluent Bit

L'Fluent Bitimage AWS pour est disponible sur :

- Amazon ECR sur la Galerie publique Amazon ECR
- Le référentiel Amazon ECR (dans la plupart des régions de haute disponibilité)

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 Fluent Bit image AWS for à l'aide de cette commande Docker CLI :

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:latest
```

Consultez également les billets de blog suivants pour plus d'informations sur Fluent Bit et les fonctionnalités associées :

- Fluent Bitpour Amazon EKS sur AWS Fargate

- Centralized Container Logging with Fluent Bit
- Création d'un agrégateur de solutions de journalisation évolutif avec AWS Fargate Fluentd et Amazon Data Firehose

Routage personnalisé des journaux - FireLens pour Amazon ECS

Avec Amazon ECS, vous pouvez utiliser les paramètres de définition des tâches FireLens 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. FireLens fonctionne avec Fluentd et. Fluent Bit Nous fournissons l'Fluent Bitimage AWS pour. Vous pouvez également utiliser votre propre Fluentd ou image Fluent Bit.

Vous devez tenir compte des conditions et considérations suivantes lors de l'utilisation FireLens d'Amazon ECS :

- FireLens pour Amazon ECS est pris en charge pour les tâches hébergées à la fois sur Amazon EC2 AWS Fargate et Amazon EC2.
- FireLens pour Amazon ECS est pris en charge dans les AWS CloudFormation modèles. Pour plus d'informations, consultez AWS::ECS::TaskDefinition FirelensConfiguration dans le Guide de l'utilisateur AWS CloudFormation .
- Pour les tâches qui utilisent le mode `bridge` réseau, les conteneurs dotés de la FireLens configuration doivent démarrer avant que les conteneurs d'applications qui en dépendent ne démarrent. Pour contrôler l'ordre de démarrage de vos conteneurs, utilisez les conditions de dépendance dans la définition de tâche. Pour plus d'informations, veuillez consulter Dépendances de conteneur (langue française non garantie).

AWS Fargate sécurité

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

Votre stockage éphémère doit être crypté par. AWS KMS Pour les tâches Amazon ECS hébergées sur une version AWS Fargate de plate-forme `1.4.0` ou ultérieure, chaque tâche reçoit 20 GiB de stockage éphémère. 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 plus tard, le stockage éphémère est chiffré à l'aide d'un algorithme de chiffrement AES-256 avec une clé de chiffrement gérée par AWS Fargate.

Pour plus d'informations, veuillez consulter Utilisation de volumes de données dans des tâches (langue française non garantie).

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

La commande suivante lancera une tâche Amazon ECS sur la version 1.4 de la AWS Fargate plateforme. Comme cette tâche est lancée dans le cadre du cluster Amazon ECS, elle utilise les 20 Gio 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

La configuration par défaut des fonctionnalités Linux ajoutées ou supprimées de votre conteneur est fournie par Docker. Pour plus d'informations sur les fonctionnalités disponibles, veuillez consulter Privilège d'exécution et fonctionnalités Linux dans la documentation Docker run (langue française non garantie).

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

Reportez-vous au didacticiel vidéo ci-dessous qui montre comment utiliser cette fonctionnalité dans le cadre du projet Sysdig Falco.

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

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

Utiliser Amazon GuardDuty 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 instances Amazon EC2 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.

AWS Fargate considérations relatives à la sécurité

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 des groupes de sécurité et des listes ACL réseau pour contrôler le trafic entrant et sortant. Les tâches Fargate reçoivent une adresse IP depuis le sous-réseau configuré de votre VPC.

Considérations relatives à la sécurité d'instance de conteneur EC2

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 réseau `awsvpc` d'accéder aux informations d'identification fournies par le profil d'instance Amazon EC2, tout en accordant les autorisations fournies par le rôle de la tâche, définissez la variable de configuration d'agent `ECS_AWSVPC_BLOCK_IMDS` 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.

Sécurité des tâches et des conteneurs

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.

Recommandations

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 constructions à plusieurs étapes, veuillez consulter Création de constructions à plusieurs étapes (langue française non garantie).

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.

Les images stockées dans Amazon ECR peuvent être analysées en mode push ou à la demande (une fois toutes les 24 heures). L'analyse de base d'Amazon ECR utilise Clair, une solution d'analyse d'images open source. L'analyse améliorée d'Amazon ECR utilise Amazon Inspector. Une fois l'image numérisée, les résultats sont enregistrés dans le flux d'événements Amazon ECR sur Amazon EventBridge. 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. SonarQubeest une option populaire pour les tests statiques de sécurité des applications (SAST), qui prend en charge différents langages de programmation.

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 pipeline CI/CD, vous devez vérifier les Dockerfiles pour rechercher la directive `USER` et faire échouer 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.

Configurez 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 attribue un minimum de deux processeurs.

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 d'exécuter des conteneurs dotés de privilèges (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 KernalCapabilities. 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.
- Reportez-vous à Fonctionnalités Linux 101 pour plus d'informations (langue française non garantie).

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.

Sécurité de l'exécution

La sécurité d'exécution fournit une protection active à vos conteneurs lorsqu'ils sont en cours d'exécution. L'idée est de détecter et d'empêcher les activités malveillantes de se produire sur vos conteneurs. La configuration de sécurité d'exécution diffère entre les conteneurs Windows et Linux.

Pour sécuriser un conteneur Microsoft Windows, veuillez consulter Sécuriser les conteneurs Windows.

Pour sécuriser un conteneur Linux, vous pouvez ajouter ou supprimer des fonctionnalités du noyau Linux à l'aide `linuxParameters` de SELinux `labels` ou d'un AppArmor profil utilisant les deux par conteneur au sein d'une définition de tâche. `dockerSecurityOptions` SELinux ou AppArmor doivent être configurés sur l'instance de conteneur avant de pouvoir être utilisés. SELinux et ne AppArmor sont pas disponibles dans. AWS Fargate Pour plus d'informations, veuillez consulter `dockerSecurityOptions` dans la Référence d'API Amazon Elastic Container Service et Configuration de sécurité dans la référence Docker run (langue française non garantie).

AppArmor est un module de sécurité Linux qui restreint les capacités d'un conteneur, notamment l'accès à certaines parties du système de fichiers. Il peut être exécuté en mode `enforcement` ou `complain`. Comme la création de AppArmor profils peut être difficile, nous vous recommandons d'utiliser un outil tel que Bane. Pour plus d'informations AppArmor, consultez la AppArmorpage officielle.

> **Important**
>
> AppArmor n'est disponible que pour les distributions Ubuntu et Debian de Linux.

Recommandations

Nous vous recommandons d'effectuer les opérations suivantes lorsque vous configurez la sécurité de votre environnement d'exécution.

Utiliser Amazon GuardDuty 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 instances Amazon EC2 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.

Utilisez une solution tierce pour la défense de l'exécution

Utilisez une solution tierce pour la défense de l'exécution. Si vous connaissez le fonctionnement de la sécurité Linux, créez et gérez AppArmor des profils. Il s'agit de projets open source. Dans le cas contraire, pensez plutôt à utiliser un autre service tiers. La plupart ont recours au machine learning pour bloquer ou signaler toute activité suspecte. Pour obtenir la liste des solutions tierces disponibles, veuillez consulter AWS Marketplace pour conteneurs (langue française non garantie).

Bonnes pratiques en matière d'AMI

Amazon ECS fournit les AMI optimisées pour Amazon ECS qui sont préconfigurées avec les exigences et les recommandations pour exécuter vos charges de travail de conteneur. Nous vous recommandons d'utiliser l'AMI Amazon Linux 2023 optimisée pour Amazon ECS pour vos instances Amazon EC2, sauf si votre application nécessite des instances Amazon EC2 basées sur un GPU, un système d'exploitation spécifique ou une version de Docker qui n'est pas encore disponible dans cette AMI. Pour plus d'informations sur les instances Amazon Linux 2 et Amazon Linux 2023, veuillez consulter Comparaison entre Amazon Linux 2 et Amazon Linux 2023 dans le Guide de l'utilisateur Amazon Linux 2023 (langue française non garantie).

Bien que vous puissiez créer votre propre AMI d'instance Amazon EC2 qui répond aux spécifications de base nécessaires pour exécuter vos charges de travail conteneurisées sur Amazon ECS, les AMI optimisées pour Amazon EC2 sont préconfigurées et testées sur Amazon ECS par des ingénieurs. AWS C'est le moyen le plus simple de démarrer et de faire fonctionner AWS rapidement vos conteneurs.

AWS Les partenaires

Vous pouvez utiliser l'un des produits AWS partenaires suivants pour ajouter une sécurité et des fonctionnalités supplémentaires à vos charges de travail Amazon ECS. Pour plus d'informations, veuillez consulter Partenaires Amazon ECS (langue française non garantie).

Aqua Security

Vous pouvez utiliser Aqua Security pour sécuriser vos applications natives cloud, du développement à la production. La solution Aqua Cloud Native Security Platform s'intègre à vos ressources natives cloud et à vos outils d'orchestration pour fournir une sécurité transparente et automatisée. Elle peut empêcher les activités suspectes et les attaques en temps réel, et contribuer à appliquer les stratégies et à simplifier la conformité réglementaire.

Palo Alto Networks

Palo Alto Networks assure la sécurité et la protection de vos hôtes, de vos conteneurs et de votre infrastructure sans serveur dans le cloud et tout au long du cycle de développement et du cycle de vie des logiciels.

Twistlock est fourni par Palo Alto Networks et peut être intégré à Amazon ECS. FireLens Grâce à elle, vous avez accès à des journaux de sécurité et à des incidents de haute fidélité qui sont regroupés de manière fluide dans plusieurs AWS services. Il s'agit notamment d'Amazon CloudWatch, Amazon Athena et Amazon Kinesis. Twistlock sécurise les charges de travail déployées sur les services de conteneurs. AWS

Sysdig

Vous pouvez utiliser Sysdig pour exécuter des charges de travail natives cloud sécurisées et conformes dans des scénarios de production. La DevOps plateforme sécurisée Sysdig intègre des fonctionnalités de sécurité et de conformité pour protéger vos charges de travail natives dans le cloud, et offre également une évolutivité, des performances et une personnalisation de niveau professionnel.

Récupérez les métadonnées Amazon ECS

Amazon ECS fournit diverses métadonnées pour votre configuration.

Vous pouvez utiliser les métadonnées pour obtenir des informations sur les tâches, notamment :

- 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

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.

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 plus d'informations, consultez Tâche automatisée et nettoyage d'image.

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 plus d'informations, consultez 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 plus d'informations, consultez 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": "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/2b88376d-
aba3-4950-9ddf-bcb0f388a40c",
    "TaskDefinitionFamily": "console-sample-app-static",
    "TaskDefinitionRevision": "1",
    "ContainerID": "aec2557997f4eed9b280c2efd7afccdcedfda4ac399f7480cae870cfc7e163fd",
    "ContainerName": "simple-app",
    "CreatedAt": "2023-10-08T20:09:11.44527186Z",
    "StartedAt": "2023-10-08T20:09:11.44527186Z",
```

```
    "DockerContainerName": "/ecs-console-sample-app-static-1-simple-app-
e4e8e495e8baa5de1a00",
    "ImageID":
 "sha256:2ae34abc2ed0a22e280d17e13f9c01aaf725688b09b7a1525d1a2750e2c0d1de",
    "ImageName": "httpd:2.4",
    "PortMappings": [
        {
            "ContainerPort": 80,
            "HostPort": 80,
            "BindIp": "0.0.0.0",
            "Protocol": "tcp"
        }
    ],
    "Networks": [
        {
            "NetworkMode": "bridge",
            "IPv4Addresses": ["192.0.2.0"]
        }
    ],
    "MetadataFileStatus": "READY",
    "AvailabilityZone": "us-east-1b",
    "HostPrivateIPv4Address": "192.0.2.0",
    "HostPublicIPv4Address": "203.0.113.0"
}
```

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 Linux Amazon EC2 exécutant au moins la version `1.39.0` de l'agent de conteneur Amazon ECS. Pour les instances Windows Amazon EC2 qui utilisent le mode réseau `awsvpc`, la version de l'agent de conteneur Amazon ECS doit être au moins `1.54.0`. Pour plus d'informations, consultez 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 Linux Amazon EC2 exécutant au moins la version `1.21.0` de l'agent de conteneur Amazon ECS. Pour les instances Windows Amazon EC2 qui utilisent le mode réseau `awsvpc`, la version de l'agent de conteneur Amazon ECS doit être au moins `1.54.0`. Pour plus d'informations, consultez Point de terminaison des métadonnées des tâches Amazon ECS, version 3.
- Point de terminaison de métadonnées de tâche version 2 : disponible pour les tâches Amazon ECS lancées sur des instances Linux Amazon EC2 exécutant au moins la version `1.17.0` de l'agent de conteneur Amazon ECS. Pour les instances Windows Amazon EC2 qui utilisent le mode réseau `awsvpc`, la version de l'agent de conteneur Amazon ECS doit être au moins `1.54.0`. Pour plus d'informations, consultez 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 de service de métadonnées d'instance (IMDS). 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, veuillez consulter 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 plus d'informations, consultez Surveillez les conteneurs Amazon ECS à l'aide de Container Insights.

> **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 les instances Linux Amazon EC2 exécutant au moins la version `1.39.0` de l'agent de conteneur Amazon ECS. Pour les instances Windows Amazon EC2 qui utilisent le mode réseau `awsvpc`, la version de l'agent de conteneur Amazon ECS doit être au moins `1.54.0`. Pour plus d'informations, consultez Gestion des instances de conteneur Linux.

> **Note**
>
> Vous pouvez ajouter la prise en charge de cette fonctionnalité sur les instances Amazon EC2 utilisant des versions plus anciennes de l'agent de conteneur Amazon ECS en mettant à jour l'agent à la dernière version. Pour plus d'informations, consultez Mise à jour de l'agent de conteneur Amazon ECS.

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 pour la tâche, y compris une liste des ID et noms de conteneur pour 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 champ `Errors` est uniquement dans la réponse pour les tâches hébergées sur des instances Linux Amazon EC2 exécutant au moins la version `1.50.0` de l'agent de conteneur. Pour les instances Windows Amazon EC2 qui utilisent le mode réseau `awsvpc`, la version de l'agent de conteneur Amazon ECS doit être au moins `1.54.0`. Ce point de terminaison nécessite l'autorisation `ecs.ListTagsForResource`.

`${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 les modes réseau `awsvpc` ou `bridge` hébergés sur des instances Linux Amazon EC2 exécutant au moins la version `1.43.0` de l'agent de conteneur,

il y aura des statistiques de débit réseau supplémentaires 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 les modes réseau `awsvpc` ou `bridge` hébergés sur des instances Linux Amazon EC2 exécutant au moins la version `1.43.0` de l'agent de conteneur, il y aura des statistiques de débit réseau supplémentaires 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'identifiant VPC de l'instance de conteneur Amazon EC2. Ce champ apparaît uniquement pour les instances Amazon EC2.

> **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`

Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`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`

Limites de ressource spécifiées au niveau de la tâche, par exemple, le processeur (exprimé en vCPU) 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 l'infrastructure Fargate ou EC2.

> **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`

Hachage SHA-256 de l'image.

`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`

Amazon Resource Name (ARN) complet 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.

`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 sortie.

```
{
    "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 les modes réseau `awsvpc` ou `bridge` hébergés sur des instances Amazon EC2 exécutant au moins la

version `1.43.0` de l'agent de conteneur, il y aura des statistiques de débit réseau supplémentaires 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 réseau `bridge`.


```
{
    "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 plus d'informations, consultez 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 dans le guide de l'utilisateur d'Amazon Elastic Container Service pour AWS 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`.

Le point de terminaison des métadonnées de tâches version 3 est activé par défaut pour les tâches qui utilisent le type de lancement Fargate sur la version de plateforme v1.3.0 ou ultérieure et les tâches qui utilisent le type de lancement EC2 et sont lancées sur l'infrastructure Linux Amazon EC2 exécutant au moins la version 1.21.0 de l'agent de conteneur Amazon ECS ou sur une infrastructure Windows Amazon EC2 exécutant au moins la version `1.54.0` de l'agent de conteneur Amazon ECS

et qui utilisent le mode réseau `awsvpc`. Pour plus d'informations, consultez Gestion des instances de conteneur Linux.

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 plus d'informations, consultez 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 plus d'informations, consultez 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 le JSON des métadonnées pour la tâche, y compris une liste des ID et noms de conteneur pour 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`

Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`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`

Limites de ressource spécifiées au niveau de la tâche, par exemple, le processeur (exprimé en vCPU) 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`

Hachage SHA-256 de l'image.

`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 l'utilisateur Amazon EC2 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 plus d'informations, consultez 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 qui sont lancées avec le mode réseau `awsvpc` reçoivent une adresse IPv4 locale au sein d'une plage d'adresses link-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 plus d'informations, consultez Versions de plateforme Linux Fargate.
- Tâches utilisant le type de lancement EC2 qui utilisent également le mode réseau `awsvpc` et qui sont lancées sur l'infrastructure Linux Amazon EC2 exécutant au moins la version 1.17.0 de l'agent de conteneur Amazon ECS ou sur l'infrastructure Windows Amazon EC2 exécutant au moins la version `1.54.0` de l'agent du conteneur Amazon ECS. Pour plus d'informations, consultez Gestion des instances de conteneur Linux.

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 plus d'informations, consultez 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 le JSON des métadonnées pour la tâche, y compris une liste des ID et noms de conteneur pour 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`

Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`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`

Limites de ressource spécifiées au niveau de la tâche, par exemple, le processeur (exprimé en vCPU) 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`

Hachage SHA-256 de l'image.

`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 l'utilisateur Amazon EC2 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 qui sont lancées avec le mode réseau `awsvpc` reçoivent une adresse IPv4 locale au sein d'une plage d'adresses link-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 instances Amazon EC2, 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 pour la tâche, y compris une liste des ID et noms de conteneur pour 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'identifiant VPC de l'instance de conteneur Amazon EC2. Ce champ apparaît uniquement pour les instances Amazon EC2.

> **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`

Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`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`

Limites de ressource spécifiées au niveau de la tâche, par exemple, le processeur (exprimé en vCPU) 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 l'infrastructure Fargate ou EC2.

> **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`

Hachage SHA-256 de l'image.

`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`

Amazon Resource Name (ARN) complet 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`.

`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 l'utilisateur Amazon EC2 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/clusterName",
  "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,
          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 plus d'informations, consultez 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 plus d'informations, consultez Versions de plateforme Linux Fargate.

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 le JSON des métadonnées pour la tâche, y compris une liste des ID et noms de conteneur pour 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`

Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`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`

Limites de ressource spécifiées au niveau de la tâche, par exemple, le processeur (exprimé en vCPU) 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`

Hachage SHA-256 de l'image.

`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 l'utilisateur Amazon EC2 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 plus d'informations, consultez 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 | python -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 des ID de conteneur Docker complets pour l'API d'introspection, et pas seulement la version courte

qui est affichée avec la commande 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"
}
```

AWS services intégrés à Amazon ECS

Amazon ECS travaille avec d'autres AWS services pour fournir des solutions supplémentaires répondant aux défis de votre entreprise. Cette rubrique identifie les services qui utilisent Amazon ECS pour ajouter des fonctionnalités ou les services utilisés par Amazon ECS pour exécuter des tâches.

Table des matières

- Utilisation d'images Amazon ECR avec Amazon ECS
- AWS Deep Learning Containers sur Amazon ECS
- Utilisation Notifications des utilisateurs AWS avec Amazon ECS

Utilisation d'images Amazon ECR avec Amazon ECS

Amazon ECR est un service de registre AWS Docker géré. Les clients peuvent utiliser la CLI Docker connue pour transférer, récupérer et gérer des images. Amazon ECR fournit un registre sécurisé, évolutif et fiable. Amazon ECR prend en charge les référentiels Docker privés dotés d'autorisations basées sur les ressources à l'aide d' AWS IAM afin que des utilisateurs spécifiques ou des instances Amazon EC2 puissent accéder aux référentiels et aux images. Les développeurs peuvent utiliser la CLI Docker pour créer et gérer des images.

Pour plus d'informations sur la création de référentiels, le transfert et la récupération d'images à partir d'Amazon ECR et la définition de contrôles d'accès sur vos référentiels, veuillez consulter le Guide de l'utilisateur du registre de conteneur Amazon Elastic.

Utilisation d'images Amazon ECR avec Amazon ECS

Vous pouvez utiliser vos images ECR avec Amazon ECS, mais vous devez satisfaire aux conditions préalables suivantes :

- Vos instances de conteneur doivent utiliser au moins la version 1.7.0 de l'agent de conteneur Amazon ECS. La dernière version de l'AMI optimisée pour Amazon ECS prend en charge les images Amazon ECR dans les définitions de tâches. Pour plus de détails, y compris les derniers ID d'AMI optimisée pour Amazon ECS, veuillez consulter Versions d'agent de conteneur Amazon ECS dans le Guide du développeur Amazon Elastic Container Service.

- Le rôle d'instance de conteneur Amazon ECS (`ecsInstanceRole`) que vous utilisez avec vos instances de conteneur doit posséder les autorisations de stratégie IAM suivantes pour Amazon ECR.

```
{
    "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.

- Dans vos définitions de tâches ECS, assurez-vous d'utiliser la dénomination `registry/repository:tag` complète pour vos images ECR. Par exemple, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`*.

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 conteneurs Deep Learning Containers sans Elastic Inference sur Amazon ECS, veuillez consulter Conteneurs Deep Learning sur Amazon ECS dans le Guide du développeur AWS Deep Learning AMI .

Les conteneurs Deep Learning Containers avec Elastic Inference sur Amazon ECS

> **Note**
>
> À compter du 15 avril 2023, AWS nous n'intégrerons pas de nouveaux clients à Amazon Elastic Inference (EI) et nous aiderons les clients actuels à migrer leurs charges de travail vers des options offrant un meilleur prix et de meilleures performances. Après le 15 avril 2023, les nouveaux clients ne pourront plus lancer d'instances avec les accélérateurs Amazon EI sur Amazon SageMaker, Amazon ECS ou Amazon EC2. Toutefois, les clients qui ont utilisé Amazon EI au moins une fois au cours des 30 derniers jours sont considérés comme des clients actuels et pourront continuer à utiliser le service.

AWS Les Deep Learning Containers fournissent un ensemble d'images Docker pour diffuser des modèles dans TensorFlow et Apache MXNet (incubation) qui tirent parti des accélérateurs Amazon Elastic Inference. Amazon ECS fournit des paramètres de définition de tâche pour attacher des accélérateurs Elastic Inference à vos conteneurs. Lorsque vous spécifiez un type d'accélérateur Elastic Inference dans votre définition de tâche, Amazon ECS gère le cycle de vie et la configuration de l'accélérateur. Le rôle lié à Amazon ECS service est requis lors de l'utilisation de cette fonction. Pour de plus amples informations sur les accélérateurs Elastic Inference, veuillez consulter Amazon Elastic Inference Basics.

> **Important**
>
> Vos instances de conteneur Amazon ECS nécessitent au minimum la version `1.30.0` 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.

Pour commencer à utiliser les conteneurs Deep Learning Containers avec Elastic Inference sur Amazon ECS, veuillez consulter Conteneurs Deep Learning avec Elastic Inference sur Amazon ECS dans le Guide du développeur Amazon Elastic Inference.

Utilisation Notifications des utilisateurs AWS avec Amazon ECS

Vous pouvez l'utiliser Notifications des utilisateurs AWSpour configurer des canaux de diffusion afin d'être informé des événements Amazon ECS. Vous recevez une notification lorsqu'un événement correspond à une règle que vous avez spécifiée. Vous pouvez recevoir des notifications relatives à des événements via plusieurs canaux, notamment des e-mails, des notifications de chat AWS Chatbot ou des notifications push AWS Console Mobile Application. Vous pouvez également consulter les notifications dans le centre de notifications de la console. Notifications des utilisateurs prend en charge l'agrégation, ce qui peut réduire le nombre de notifications que vous recevez lors d'événements spécifiques.

Exemple

Le modèle d'événement suivant correspond à un changement d'état de tâche sur le cluster nommé « par défaut ».

```
{
  "source": ["aws.ecs"],
  "detail-type": ["ECS Task State Change"]
  "detail": {
    "clusterArn": [
      "default"
    ]
  }
}
```

Didacticiels pour Amazon ECS

Les didacticiels suivants montrent comment exécuter les tâches courantes à l'aide d'Amazon ECS.

Rubriques

- Utilisation de conteneurs Windows sur AWS Fargate
- Création d'un cluster avec une tâche Fargate Linux à l'aide du AWS CLI
- Création d'un cluster avec une tâche Fargate Windows à l'aide du AWS CLI
- Création d'un cluster avec une tâche EC2 à l'aide du AWS CLI
- Utilisation de la mise à l'échelle automatique du cluster avec la console AWS Management Console et la console Amazon ECS
- Spécification de données sensibles à l'aide de secrets de Secrets Manager
- Création d'un service à l'aide de Service Discovery
- Création d'un service à l'aide d'un déploiement bleu/vert
- Écouter les CloudWatch événements Amazon ECS
- Envoi d'alertes Amazon Simple Notification Service en cas d'arrêt d'une tâche
- Concaténation des messages du journal multiligne ou de suivi de pile
- Utilisation des systèmes de fichiers Amazon EFS avec Amazon ECS à l'aide de la console
- Utilisation de systèmes de fichiers FSx for Windows File Server avec Amazon ECS
- Déploiement de Fluent Bit sur des conteneurs Amazon ECS pour Windows
- Utilisation des GMSA pour les conteneurs Windows sur Amazon EC2
- Utilisation de conteneurs Windows avec Domainless à gMSA l'aide du AWS CLI
- Utilisation gMSA pour les Linux conteneurs sur Amazon EC2
- Utilisation gMSA pour les Linux conteneurs sur Fargate
- Utilisation d'EC2 Image Builder pour créer des AMI personnalisées optimisées pour Amazon ECS

Utilisation de conteneurs Windows sur AWS 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 AWS utilisateur 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 plus d'informations, consultez 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 plus d'informations, consultez 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, veuillez consulter Ajout de règles à votre groupe de sécurité dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux (langue française non garantie).

- 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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.

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 instances Amazon EC2, les équilibreurs de charge Elastic Load Balancing et les groupes Auto Scaling, doivent être nettoyées manuellement dans la console Amazon EC2 ou en supprimant la pile qui les AWS CloudFormation 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 delete ***cluster-name***, puis choisissez Delete (Supprimer). La suppression du cluster nettoie les ressources associées qui ont été créées avec le cluster, y compris les groupes Auto Scaling, les VPC ou les équilibreurs de charge.

Création d'un cluster avec une tâche Fargate Linux à l'aide du 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. Veillez à utiliser la dernière version de l' AWS CLI. Pour savoir comment opérer une mise à niveau vers la dernière version, consultez Installation de AWS Command Line Interface.

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 AWS Command Line Interface.
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre AWS utilisateur 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 plus d'informations, consultez the section called "Créer un Virtual Private Cloud".

Pour de plus amples informations sur les groupes de sécurité et les règles, veuillez consulter Groupes de sécurité par défaut de vos VPC et Exemples de règles dans le Guide de l'utilisateur 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, veuillez consulter Utilisation d'Amazon ECS Exec pour le débogage (langue française non garantie).
- (Facultatif) AWS CloudShell est un outil qui fournit aux clients une ligne de commande sans avoir à créer leur propre instance EC2. 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'un cluster avec une tâche Fargate Windows à l'aide du 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 savoir comment opérer une mise à niveau vers la dernière version, consultez Installation de AWS Command Line Interface.

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 Command Line Interface.
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre AWS utilisateur 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 plus d’informations, consultez the section called “Créer un Virtual Private Cloud”.

  Pour de plus amples informations sur les groupes de sécurité et les règles, veuillez consulter Groupes de sécurité par défaut de vos VPC et Exemples de règles dans le Guide de l'utilisateur Amazon Virtual Private Cloud.
- (Facultatif) AWS CloudShell est un outil qui fournit aux clients une ligne de commande sans avoir à créer leur propre instance EC2. 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'un cluster avec une tâche EC2 à l'aide du 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. Veillez à utiliser la dernière version de l' AWS CLI. Pour savoir comment opérer une mise à niveau vers la dernière version, consultez Installation de AWS Command Line Interface.

Rubriques

- Prérequis
- Étape 1 : Créer un cluster
- Étape 2 : Lancer une instance avec l'AMI Amazon ECS
- Étape 3 : Répertorier les instances de conteneur
- Étape 4 : Décrire votre instance de conteneur
- Étape 5 : Enregistrer une définition de tâche
- Étape 6 : Répertorier les définitions de tâche
- Étape 7 : Exécuter une tâche
- Étape 8 : Répertorier les tâches
- Étape 9 : Décrire la tâche en cours d'exécution

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 Command Line Interface.
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.

- Votre AWS utilisateur 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. Pour plus d'informations, consultez 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 instance EC2. 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` 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"
    }
}
```

Étape 2 : Lancer une instance avec l'AMI Amazon ECS

Vous devez disposer d'une instance de conteneur Amazon ECS dans votre cluster avant de pouvoir y exécuter des tâches. Si vous ne disposez pas d'instances de conteneur dans votre cluster, consultez Lancement d'une instance de conteneur Amazon ECS Linux pour plus d'informations.

Étape 3 : Répertorier 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"
    ]
}
```

Étape 4 : Décrire 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 default --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'instance Amazon EC2 que vous pouvez utiliser pour surveiller l'instance dans la console Amazon EC2 ou avec la commande aws ec2 describe-instances --instance-id ***instance_id***.

Étape 5 : Enregistrer une définition de tâche

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 utilise une image `busybox` de Docker Hub et qui reste simplement en veille pendant 360 secondes. Pour plus d'informations sur les paramètres de définition des tâches disponibles, consultez Définitions de tâche Amazon ECS.

```
{
    "containerDefinitions": [
        {
            "name": "sleep",
            "image": "busybox",
            "cpu": 10,
            "command": [
                "sleep",
                "360"
            ],
            "memory": 10,
```

```
            "essential": true
        }
    ],
    "family": "sleep360"
}
```

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 utiliser un caractère d'échappement pour les guillemets dans le fichier JSON et transmettre les définitions de conteneur JSON sur la ligne de commande comme dans l'exemple ci-dessous. 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/sleep360.json
```

Pour utiliser une chaîne JSON pour les définitions de conteneur :

```
aws ecs register-task-definition --family sleep360 --container-definitions "[{\"name
\":\"sleep\",\"image\":\"busybox\",\"cpu\":10,\"command\":[\"sleep\",\"360\"],\"memory
\":10,\"essential\":true}]"
```

La commande register-task-definition renvoie une description de la définition de tâche une fois l'enregistrement terminé.

```
{
    "taskDefinition": {
        "volumes": [],
        "taskDefinitionArn": "arn:aws:ec2:us-east-1:aws_account_id:task-definition/
sleep360:1",
        "containerDefinitions": [
            {
                "environment": [],
                "name": "sleep",
                "mountPoints": [],
                "image": "busybox",
                "cpu": 10,
```

```
                "portMappings": [],
                "command": [
                    "sleep",
                    "360"
                ],
                "memory": 10,
                "essential": true,
                "volumesFrom": []
            }
        ],
        "family": "sleep360",
        "revision": 1
    }
}
```

Étape 6 : 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:ec2:us-east-1:aws_account_id:task-definition/sleep300:1",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep300:2",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep360:1",
        "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"
    ]
}
```

Étape 7 : Exécuter une tâche

Une fois que vous avez enregistré une tâche pour votre compte et que vous avez lancé une instance de conteneur qui est enregistrée sur votre cluster, vous pouvez exécuter la tâche enregistrée

dans votre cluster. Pour cet exemple, vous placez une seule instance de la définition de tâche `sleep360:1` dans votre cluster par défaut.

```
aws ecs run-task --cluster default --task-definition sleep360:1 --count 1
```

Sortie :

```
{
    "tasks": [
        {
            "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID",
            "overrides": {
                "containerOverrides": [
                    {
                        "name": "sleep"
                    }
                ]
            },
            "lastStatus": "PENDING",
            "containerInstanceArn": "arn:aws:ecs:us-east-1:aws_account_id:container-
instance/container_instance_ID",
            "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/default",
            "desiredStatus": "RUNNING",
            "taskDefinitionArn": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/
sleep360:1",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-
east-1:aws_account_id:container/container_ID",
                    "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID",
                    "lastStatus": "PENDING",
                    "name": "sleep"
                }
            ]
        }
    ]
}
```

Étape 8 : Répertorier les tâches

Affichez les tâches de votre cluster. Vous devez voir la tâche que vous avez exécutée dans la section précédente. Vous pouvez noter l'ID de la tâche ou l'ARN complet qui est renvoyé par cette commande et l'utiliser pour décrire la tâche ultérieurement.

```
aws ecs list-tasks --cluster default
```

Sortie :

```
{
    "taskArns": [
        "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID"
    ]
}
```

Étape 9 : Décrire la tâche en cours d'exécution

Décrivez la tâche à l'aide de l'ID de tâche récupérée précédemment afin d'obtenir plus d'informations sur la tâche.

```
aws ecs describe-tasks --cluster default --task task_ID
```

Sortie :

```
{
    "failures": [],
    "tasks": [
        {
            "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID",
            "overrides": {
                "containerOverrides": [
                    {
                        "name": "sleep"
                    }
                ]
            },
            "lastStatus": "RUNNING",
            "containerInstanceArn": "arn:aws:ecs:us-east-1:aws_account_id:container-
instance/container_instance_ID",
```

```
            "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/default",
            "desiredStatus": "RUNNING",
            "taskDefinitionArn": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/
sleep360:1",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-
east-1:aws_account_id:container/container_ID",
                    "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID",
                    "lastStatus": "RUNNING",
                    "name": "sleep",
                    "networkBindings": []
                }
            ]
        }
    ]
}
```

Utilisation de la mise à l'échelle automatique du cluster avec la console AWS Management Console et la console Amazon ECS

Ce didacticiel vous explique comment créer les ressources pour Cluster Auto Scaling à l'aide de la AWS Management Console. Quand les ressources devront être nommées, nous utiliserons le préfixe `ConsoleTutorial` de sorte qu'elles aient toutes un nom unique et qu'il soit facile de les retrouver.

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 AWS utilisateur 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 plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.
- Le rôle IAM lié à un service Amazon ECS service est créé. Pour plus d'informations, consultez 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 Rôles liés à un service pour Amazon EC2 Auto Scaling dans le guide de l'utilisateur Amazon EC2 Auto Scaling.
- Vous avez un créé un VPC et un groupe de sécurité prêts à être utilisés. Pour plus d'informations, consultez 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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 les instances Amazon 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.
        - Pour EC2 instance type (Type d'instance EC2), 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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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": "amazonlinux:2",
            "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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 console Amazon EC2 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.

Effectuez les étapes suivantes pour examiner votre cluster Amazon ECS et vérifier que les instances Amazon EC2 ont bien été enregistrées auprès du cluster et que vos tâches sont passées à l'état `RUNNING`.

Vérifier les instances dans le groupe Auto Scaling

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 console Amazon EC2 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.

Spécification de données sensibles à l'aide de secrets de Secrets Manager

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 plus d'informations, consultez Transmission de données sensibles à un conteneur.

Le didacticiel suivant montre comment créer un secret Secrets Manager, référencer le secret dans une définition de tâche Amazon ECS, puis vérifier que cela fonctionné en interrogeant la variable d'environnement dans un conteneur montrant 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 AWS utilisateur dispose des autorisations IAM requises pour créer les ressources Secrets Manager et Amazon ECS décrites.

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, consultez Didacticiel : création et récupération d'un secret dans le Guide de l'utilisateur AWS Secrets Manager .

Les paires clé/valeur 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 : Mettre à jour votre rôle IAM d'exécution de tâche

Pour permettra à Amazon ECS de récupérer les données sensibles à partir de votre secret Secrets Manager, vous devez avoir le rôle d'exécution de tâche Amazon ECS et y faire référence dans votre définition de tâche. Cela permet à l'agent de conteneur d'extraire les ressources Secrets Manager

nécessaires. Si vous n'avez pas encore créé votre rôle d'exécution de tâche IAM, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Les étapes suivantes impliquent que vous ayez déjà créé et correctement configuré le rôle IAM d'exécution de tâche.

Pour mettre à jour votre rôle IAM d'exécution de tâche

Utilisez la console IAM pour mettre à jour votre rôle d'exécution de tâche avec les autorisations requises.

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le panneau de navigation, sélectionnez Rôles.
3. Recherchez `ecsTaskExecutionRole` dans la liste des rôles et sélectionnez-le.
4. Choisissez Permissions (Autorisations), Add inline policy (Ajouter une politique en ligne).
5. Choisissez l'onglet JSON et saisissez le texte JSON suivant, en veillant à spécifier l'ARN complet du secret Secrets Manager que vous avez créé à l'étape 1.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [

 "arn:aws:secretsmanager:region:aws_account_id:secret:username_value"
            ]
        }
    ]
}
```

6. Choisissez Review policy (Examiner une politique). Dans Name (Nom), spécifiez `ECSSecretsTutorial`, puis choisissez Create policy (Créer une politique).

Étape 3 : Créer une définition de tâche Amazon ECS

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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, saisissez le texte JSON de définition de tâche suivant, en veillant à spécifier l'ARN complet du secret Secrets Manager que vous avez créée à l'étape 1 et le rôle IAM 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": "httpd:2.4",
            "essential": true,
            "name": "ecs-secrets-container"
        }
    ],
    "family": "ecs-secrets-tutorial"
}
```

6. Choisissez Créer.

Étape 4 : Créer un cluster Amazon ECS

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 lancement EC2, consultez la section the section called “Création d'un cluster pour le type de lancement Amazon EC2 à l'aide de la console”.

Étape 5 : Exécuter une tâche Amazon ECS

Vous pouvez utiliser la console Amazon ECS pour exécuter une tâche avec la définition de tâche que vous avez créée. Dans le cadre de ce didacticiel, nous allons exécuter une tâche à l'aide du type de lancement EC2, à l'aide du cluster que nous avons créé lors de l'étape précédente.

Pour plus d'informations sur l'exécution d’une tâche, consultez the section called “Exécuter une application en tant que tâche Amazon ECS”.

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
   b. Dans le volet de navigation, choisissez Clusters, puis sélectionnez 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, veuillez consulter Se connecter à votre instance Linux à partir de Windows à l'aide de PuTTY dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

   > **Important**
   >
   > Pour plus d'informations sur les problèmes que vous pouvez rencontrer lors de la connexion à votre instance, veuillez consulter Résolution de la connexion à votre instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 delete ***nom du cluster***, puis choisissez Supprimer.
6. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
7. Dans le panneau de navigation, sélectionnez Rôles.
8. Recherchez `ecsTaskExecutionRole` dans la liste des rôles et sélectionnez-le.
9. Choisissez Permissions, puis le X à côté d'ECS SecretsTutorial. 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).

Création d'un service à l'aide de Service Discovery

La découverte de service est désormais intégrée dans l'assistant de création de service da la console Amazon ECS. Pour plus d'informations, consultez Création d'un service à l'aide de la console.

Le didacticiel suivant explique comment créer un service ECS à l'aide d'une tâche Fargate qui utilise la découverte de service avec l' AWS CLI.

Pour obtenir la liste de Régions AWS ces services d'assistance découverts, consultezDécouverte de service.

Pour de plus amples informations sur les Régions qui prennent en charge Fargate, veuillez consulter the section called "AWS Régions de Fargate".

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).
- Les étapes décrites dans Configurer l'utilisation d'Amazon ECS sont terminées.
- Votre AWS utilisateur 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 plus d'informations, consultez 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`. Remplacez *`vpc-abcd1234`* avec l'ID de l'un de vos VPC existants.

```
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": "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 les ID du groupe de sécurité et du sous-réseau à partir de 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
```

Création d'un service à l'aide d'un déploiement bleu/vert

Le didacticiel suivant explique comment créer un Amazon ECS service contenant une tâche Fargate qui utilise le type de déploiement bleu/vert avec l' AWS CLI.

> **Note**
>
> La prise en charge d'un déploiement bleu/vert a été ajoutée pour AWS CloudFormation. Pour plus d'informations, consultez la section Effectuer des déploiements bleu/vert d'Amazon ECS AWS CloudFormation dans le AWS CloudFormation guide CodeDeploy de l'utilisateur.

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 du AWS CLI, voir Installation du AWS Command Line Interface.
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre AWS utilisateur 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. Pour plus d’informations, consultez the section called “Créer un Virtual Private Cloud”.

- Le rôle Amazon ECS CodeDeploy IAM est créé. Pour plus d'informations, consultez Rôle CodeDeploy IAM d'Amazon ECS.

Étape 1 : Créer un Application Load Balancer

Les services Amazon ECS service utilisant le type de déploiement bleu/vert nécessitent l'utilisation d'un Application Load Balancer ou d'un Network Load Balancer. Ce didacticiel utilise un équilibreur de charge Application Load Balancer.

Pour créer un Application Load Balancer

1. Utilisez la create-load-balancercommande pour créer un Application Load Balancer. Spécifiez deux sous-réseaux qui ne proviennent pas de la même zone de disponibilité, ainsi qu'un groupe de sécurité.

   ```
   aws elbv2 create-load-balancer \
        --name bluegreen-alb \
        --subnets subnet-abcd1234 subnet-abcd5678 \
        --security-groups sg-abcd1234 \
        --region us-east-1
   ```

   Les données de sortie contiennent l'Amazon Resource Name (ARN) de l'équilibreur de charge, au format suivant :

   ```
   arn:aws:elasticloadbalancing:region:aws_account_id:loadbalancer/app/bluegreen-alb/
   e5ba62739c16e642
   ```

2. Utilisez la create-target-groupcommande pour créer un groupe cible. Ce groupe cible achemine le trafic vers la tâche d'origine définie dans votre service.

   ```
   aws elbv2 create-target-group \
        --name bluegreentarget1 \
        --protocol HTTP \
        --port 80 \
        --target-type ip \
        --vpc-id vpc-abcd1234 \
        --region us-east-1
   ```

   Les données de sortie contiennent l'ARN du groupe cible, au format suivant :

```
arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget1/209a844cd01825a4
```

3. Utilisez la commande create-listener pour créer un écouteur d'équilibreur de charge avec une règle par défaut qui transfère les demandes au groupe cible :

```
aws elbv2 create-listener \
     --load-balancer-arn
 arn:aws:elasticloadbalancing:region:aws_account_id:loadbalancer/app/bluegreen-alb/
e5ba62739c16e642 \
     --protocol HTTP \
     --port 80 \
     --default-actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup
bluegreentarget1/209a844cd01825a4 \
     --region us-east-1
```

Les données de sortie contiennent l'ARN de l'écouteur, au format suivant :

```
arn:aws:elasticloadbalancing:region:aws_account_id:listener/app/bluegreen-alb/
e5ba62739c16e642/665750bec1b03bd4
```

Étape 2 : créer un cluster Amazon ECS

Utilisez la commande create-cluster pour créer un cluster nommé `tutorial-bluegreen-cluster` à utiliser.

```
aws ecs create-cluster \
     --cluster-name tutorial-bluegreen-cluster \
     --region us-east-1
```

Les données de sortie contiennent l'ARN du cluster, au format suivant :

```
arn:aws:ecs:region:aws_account_id:cluster/tutorial-bluegreen-cluster
```

Étape 3 : Enregistrer une définition de tâche

Utilisez la register-task-definitioncommande pour enregistrer une définition de tâche compatible avec Fargate. Cela exige d'utiliser le mode réseau `awsvpc`. Voici l'exemple de définition de tâche utilisé pour ce didacticiel.

Pour commencer, créez un fichier nommé `fargate-task.json` avec le contenu suivant. Assurez-vous d'utiliser l'ARN de votre rôle d'exécution de tâche. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

```
{
    "family": "tutorial-task-def",
        "networkMode": "awsvpc",
        "containerDefinitions": [
            {
                "name": "sample-app",
                "image": "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: #00FFFF;} </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",
        "executionRoleArn": "arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole"
}
```

Enregistrez ensuite la définition de tâche à l'aide du fichier `fargate-task.json` que vous avez créé.

```
aws ecs register-task-definition \
     --cli-input-json file://fargate-task.json \
     --region us-east-1
```

Étape 4 : créer un service Amazon ECS service

Utilisez la commande create-service pour créer un service.

Pour commencer, créez un fichier nommé `service-bluegreen.json` avec le contenu suivant.

```
{
    "cluster": "tutorial-bluegreen-cluster",
    "serviceName": "service-bluegreen",
    "taskDefinition": "tutorial-task-def",
    "loadBalancers": [
        {
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget1/209a844cd01825a4",
            "containerName": "sample-app",
            "containerPort": 80
        }
    ],
    "launchType": "FARGATE",
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "CODE_DEPLOY"
    },
    "platformVersion": "LATEST",
    "networkConfiguration": {
       "awsvpcConfiguration": {
          "assignPublicIp": "ENABLED",
          "securityGroups": [ "sg-abcd1234" ],
          "subnets": [ "subnet-abcd1234", "subnet-abcd5678" ]
       }
    },
```

```
    "desiredCount": 1
}
```

Ensuite, créez votre service à l'aide du fichier `service-bluegreen.json` que vous avez créé.

```
aws ecs create-service \
     --cli-input-json file://service-bluegreen.json \
     --region us-east-1
```

Les données de sortie contiennent l'ARN du service, au format suivant :

```
arn:aws:ecs:region:aws_account_id:service/service-bluegreen
```

Obtenez le nom DNS de l'équilibreur de charge à l'aide de la commande suivante.

```
 aws elbv2 describe-load-balancers --name bluegreen-alb  --query
 'LoadBalancers[*].DNSName'
```

Saisissez le nom DNS dans votre navigateur web. Vous devriez voir une page Web qui affiche l'exemple d'application avec un fond bleu.

Étape 5 : Créer les ressources AWS CodeDeploy

Suivez les étapes ci-dessous pour créer votre CodeDeploy application, le groupe cible Application Load Balancer pour le groupe de CodeDeploy déploiement et le groupe de CodeDeploy déploiement.

Pour créer des CodeDeploy ressources

1. Utilisez la commande create-application pour créer une CodeDeploy application. Précisez la plateforme de calcul `ECS`.

   ```
   aws deploy create-application \
        --application-name tutorial-bluegreen-app \
        --compute-platform ECS \
        --region us-east-1
   ```

   Les données de sortie contiennent l'ID d'application, au format suivant :

   ```
   {
       "applicationId": "b8e9c1ef-3048-424e-9174-885d7dc9dc11"
   ```

```
}
```

2. Utilisez la create-target-groupcommande pour créer un deuxième groupe cible Application Load Balancer, qui sera utilisé lors de la création de votre groupe de CodeDeploy déploiement.

```
aws elbv2 create-target-group \
     --name bluegreentarget2 \
     --protocol HTTP \
     --port 80 \
     --target-type ip \
     --vpc-id "vpc-0b6dd82c67d8012a1" \
     --region us-east-1
```

Les données de sortie contiennent l'ARN du groupe cible, au format suivant :

```
arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget2/708d384187a3cfdc
```

3. Utilisez la create-deployment-groupcommande pour créer un groupe CodeDeploy de déploiement.

Pour commencer, créez un fichier nommé `tutorial-deployment-group.json` avec le contenu suivant. Cet exemple utilise la ressource que vous avez créée. Pour le`serviceRoleArn`, spécifiez l'ARN de votre rôle Amazon ECS CodeDeploy IAM. Pour plus d'informations, consultez Rôle CodeDeploy IAM d'Amazon ECS.

```
{
   "applicationName": "tutorial-bluegreen-app",
   "autoRollbackConfiguration": {
      "enabled": true,
      "events": [ "DEPLOYMENT_FAILURE" ]
   },
   "blueGreenDeploymentConfiguration": {
      "deploymentReadyOption": {
         "actionOnTimeout": "CONTINUE_DEPLOYMENT",
         "waitTimeInMinutes": 0
      },
      "terminateBlueInstancesOnDeploymentSuccess": {
         "action": "TERMINATE",
         "terminationWaitTimeInMinutes": 5
      }
   },
```

```
   "deploymentGroupName": "tutorial-bluegreen-dg",
   "deploymentStyle": {
      "deploymentOption": "WITH_TRAFFIC_CONTROL",
      "deploymentType": "BLUE_GREEN"
   },
   "loadBalancerInfo": {
      "targetGroupPairInfoList": [
        {
          "targetGroups": [
             {
                 "name": "bluegreentarget1"
             },
             {
                 "name": "bluegreentarget2"
             }
          ],
          "prodTrafficRoute": {
              "listenerArns": [
                  "arn:aws:elasticloadbalancing:region:aws_account_id:listener/
app/bluegreen-alb/e5ba62739c16e642/665750bec1b03bd4"
              ]
          }
        }
      ]
   },
   "serviceRoleArn": "arn:aws:iam::aws_account_id:role/ecsCodeDeployRole",
   "ecsServices": [
       {
           "serviceName": "service-bluegreen",
           "clusterName": "tutorial-bluegreen-cluster"
       }
   ]
}
```

Créez ensuite le groupe CodeDeploy de déploiement.

```
aws deploy create-deployment-group \
     --cli-input-json file://tutorial-deployment-group.json \
     --region us-east-1
```

Les données de sortie contiennent l'ID du groupe de déploiement, au format suivant :

```
{
    "deploymentGroupId": "6fd9bdc6-dc51-4af5-ba5a-0a4a72431c88"
}
```

Étape 6 : Création et surveillance d'un CodeDeploy déploiement

Avant de créer un CodeDeploy déploiement, mettez à jour la définition `command` de la tâche `fargate-task.json` comme suit pour remplacer la couleur d'arrière-plan de l'exemple d'application par le vert.

```
{
    ...
        "containerDefinitions": [
            {
                ...
                "command": [
                    "/bin/sh -c \"echo '<html> <head> <title>Amazon ECS Sample
 App</title> <style>body {margin-top: 40px; background-color: #097969;} </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\""
                ]
            }
        ],
       ...
}
```

Enregistrez la définition de tâche mise à jour à l'aide de la commande suivante.

```
aws ecs register-task-definition \
     --cli-input-json file://fargate-task.json \
     --region us-east-1
```

Procédez maintenant comme suit pour créer et télécharger un fichier de spécification d'application (AppSpec fichier) et un CodeDeploy déploiement.

Pour créer et surveiller un CodeDeploy déploiement

1. Créez et chargez un AppSpec fichier en suivant les étapes ci-dessous.

   a. Créez un fichier nommé `appspec.yaml` avec le contenu du groupe de déploiement CodeDeploy . Cet exemple utilise la définition de tâche mise à jour.

   ```
   version: 0.0
   Resources:
     - TargetService:
         Type: AWS::ECS::Service
         Properties:
           TaskDefinition: "arn:aws:ecs:region:aws_account_id:task-
   definition/tutorial-task-def:2"
           LoadBalancerInfo:
             ContainerName: "sample-app"
             ContainerPort: 80
           PlatformVersion: "LATEST"
   ```

   b. Utilisez la commande s3 mb pour créer un compartiment Amazon S3 pour le AppSpec fichier.

   ```
   aws s3 mb s3://tutorial-bluegreen-bucket
   ```

   c. Utilisez la commande s3 cp pour charger le AppSpec fichier dans le compartiment Amazon S3.

   ```
   aws s3 cp ./appspec.yaml s3://tutorial-bluegreen-bucket/appspec.yaml
   ```

2. Créez le CodeDeploy déploiement en suivant les étapes ci-dessous.

   a. Créez un fichier nommé `create-deployment.json` avec le contenu du CodeDeploy déploiement. Cet exemple utilise les ressources que vous avez créées précédemment dans le didacticiel.

   ```
   {
       "applicationName": "tutorial-bluegreen-app",
       "deploymentGroupName": "tutorial-bluegreen-dg",
       "revision": {
           "revisionType": "S3",
           "s3Location": {
               "bucket": "tutorial-bluegreen-bucket",
   ```

```
            "key": "appspec.yaml",
            "bundleType": "YAML"
        }
    }
}
```

b. Utilisez la commande create-deployment pour créer le déploiement.

```
aws deploy create-deployment \
     --cli-input-json file://create-deployment.json \
     --region us-east-1
```

Les données de sortie contiennent l'ID du déploiement, au format suivant :

```
{
    "deploymentId": "d-RPCR1U3TW"
}
```

3. Utilisez la get-deployment-targetcommande pour obtenir les détails du déploiement, en spécifiant ceux `deploymentId` de la sortie précédente.

```
aws deploy get-deployment-target \
--deployment-id "d-IMJU3A8TW" \
--target-id tutorial-bluegreen-cluster:service-bluegreen \
--region us-east-1
```

Au départ, l'état du déploiement est `InProgress`. Le trafic est dirigé vers l'ensemble de tâches d'origine, dont le `taskSetLabel` est BLUE, le statut est PRIMARY et le `trafficWeight` est `100.0`. L'ensemble de tâches de remplacement possède un `taskSetLabel` de GREEN, un statut de ACTIVE et un `trafficWeight` de `0.0`. Le navigateur Web dans lequel vous avez saisi le nom DNS affiche toujours l'exemple d'application sur fond bleu.

```
{
"deploymentTarget": {
"deploymentTargetType": "ECSTarget",
"ecsTarget": {
    "deploymentId": "d-RPCR1U3TW",
    "targetId": "tutorial-bluegreen-cluster:service-bluegreen",
    "targetArn": "arn:aws:ecs:region:aws_account_id:service/service-bluegreen",
    "lastUpdatedAt": "2023-08-10T12:07:24.797000-05:00",
    "lifecycleEvents": [
```

```
        {
            "lifecycleEventName": "BeforeInstall",
            "startTime": "2023-08-10T12:06:22.493000-05:00",
            "endTime": "2023-08-10T12:06:22.790000-05:00",
            "status": "Succeeded"
        },
        {
            "lifecycleEventName": "Install",
            "startTime": "2023-08-10T12:06:22.936000-05:00",
            "status": "InProgress"
        },
        {
            "lifecycleEventName": "AfterInstall",
            "status": "Pending"
        },
        {
            "lifecycleEventName": "BeforeAllowTraffic",
            "status": "Pending"
        },
        {
            "lifecycleEventName": "AllowTraffic",
            "status": "Pending"
        },
        {
            "lifecycleEventName": "AfterAllowTraffic",
            "status": "Pending"
        }
    ],
    "status": "InProgress",
    "taskSetsInfo": [
        {
            "identifer": "ecs-svc/9223370493423413672",
            "desiredCount": 1,
            "pendingCount": 0,
            "runningCount": 1,
            "status": "ACTIVE",
            "trafficWeight": 0.0,
            "targetGroup": {
                "name": "bluegreentarget2"
            },
            "taskSetLabel": "Green"
        },
        {
            "identifer": "ecs-svc/9223370493425779968",
```

```
            "desiredCount": 1,
            "pendingCount": 0,
            "runningCount": 1,
            "status": "PRIMARY",
            "trafficWeight": 100.0,
            "targetGroup": {
                "name": "bluegreentarget1"
            },
            "taskSetLabel": "Blue"
        }
    ]
}
}
}
```

Continuez de récupérer les détails du déploiement à l'aide de la commande jusqu'à ce que le statut de déploiement soit `Succeeded`, comme indiqué dans la sortie suivante. Le trafic est désormais redirigé vers l'ensemble de tâches de remplacement, dont le statut est désormais `PRIMARY` et le `trafficWeight` est `100.0`. Actualisez le navigateur Web dans lequel vous avez saisi le nom DNS de l'équilibreur de charge, et vous devriez maintenant voir l'exemple d'application sur fond vert.

```
{
"deploymentTarget": {
"deploymentTargetType": "ECSTarget",
"ecsTarget": {
    "deploymentId": "d-RPCR1U3TW",
    "targetId": "tutorial-bluegreen-cluster:service-bluegreen",
    "targetArn": "arn:aws:ecs:region:aws_account_id:service/service-bluegreen",
    "lastUpdatedAt": "2023-08-10T12:07:24.797000-05:00",
    "lifecycleEvents": [
        {
            "lifecycleEventName": "BeforeInstall",
            "startTime": "2023-08-10T12:06:22.493000-05:00",
            "endTime": "2023-08-10T12:06:22.790000-05:00",
            "status": "Succeeded"
        },
        {
            "lifecycleEventName": "Install",
            "startTime": "2023-08-10T12:06:22.936000-05:00",
            "endTime": "2023-08-10T12:08:25.939000-05:00",
            "status": "Succeeded"
```

```
        },
        {
            "lifecycleEventName": "AfterInstall",
            "startTime": "2023-08-10T12:08:26.089000-05:00",
            "endTime":  "2023-08-10T12:08:26.403000-05:00",
            "status": "Succeeded"
        },
        {
            "lifecycleEventName": "BeforeAllowTraffic",
            "startTime": "2023-08-10T12:08:26.926000-05:00",
            "endTime":  "2023-08-10T12:08:27.256000-05:00",
            "status": "Succeeded"
        },
        {
            "lifecycleEventName": "AllowTraffic",
            "startTime": "2023-08-10T12:08:27.416000-05:00",
            "endTime": "2023-08-10T12:08:28.195000-05:00",
            "status": "Succeeded"
        },
        {
            "lifecycleEventName": "AfterAllowTraffic",
            "startTime": "2023-08-10T12:08:28.715000-05:00",
            "endTime":  "2023-08-10T12:08:28.994000-05:00",
            "status": "Succeeded"
        }
    ],
    "status": "Succeeded",
    "taskSetsInfo": [
        {
            "identifer": "ecs-svc/9223370493425779968",
            "desiredCount": 1,
            "pendingCount": 0,
            "runningCount": 1,
            "status": "ACTIVE",
            "trafficWeight": 0.0,
            "targetGroup": {
                "name": "bluegreentarget1"
            },
            "taskSetLabel": "Blue"
        },
        {
            "identifer": "ecs-svc/9223370493423413672",
            "desiredCount": 1,
            "pendingCount": 0,
```

```
            "runningCount": 1,
            "status": "PRIMARY",
            "trafficWeight": 100.0,
            "targetGroup": {
                "name": "bluegreentarget2"
            },
            "taskSetLabel": "Green"
        }
    ]
}
}
}
```

Étape 7 : 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.

Nettoyage des ressources du didacticiel

1. Utilisez la delete-deployment-groupcommande pour supprimer le groupe CodeDeploy de déploiement.

```
aws deploy delete-deployment-group \
     --application-name tutorial-bluegreen-app \
     --deployment-group-name tutorial-bluegreen-dg \
     --region us-east-1
```

2. Utilisez la commande delete-application pour supprimer l' CodeDeploy application.

```
aws deploy delete-application \
     --application-name tutorial-bluegreen-app \
     --region us-east-1
```

3. Utilisez la commande delete-service pour supprimer l'Amazon ECS service. Utiliser l'indicateur `--force` vous permet de supprimer un service, même s'il n'a pas été abaissé à 0 tâche.

```
aws ecs delete-service \
     --service arn:aws:ecs:region:aws_account_id:service/service-bluegreen \
     --force \
```

```
    --region us-east-1
```

4. Utilisez la commande delete-cluster pour supprimer le cluster Amazon ECS.

```
aws ecs delete-cluster \
    --cluster tutorial-bluegreen-cluster \
    --region us-east-1
```

5. Utilisez la commande s3 rm pour supprimer le AppSpec fichier du compartiment Amazon S3.

```
aws s3 rm s3://tutorial-bluegreen-bucket/appspec.yaml
```

6. Utilisez la commande s3 rb pour supprimer le compartiment Amazon S3.

```
aws s3 rb s3://tutorial-bluegreen-bucket
```

7. Utilisez la delete-load-balancercommande pour supprimer l'Application Load Balancer.

```
aws elbv2 delete-load-balancer \
    --load-balancer-arn
 arn:aws:elasticloadbalancing:region:aws_account_id:loadbalancer/app/bluegreen-alb/
e5ba62739c16e642 \
    --region us-east-1
```

8. Utilisez la delete-target-groupcommande pour supprimer les deux groupes cibles Application Load Balancer.

```
aws elbv2 delete-target-group \
    --target-group-arn
 arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget1/209a844cd01825a4 \
    --region us-east-1
```

```
aws elbv2 delete-target-group \
    --target-group-arn
 arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget2/708d384187a3cfdc \
    --region us-east-1
```

Écouter les CloudWatch événements Amazon ECS

Dans ce didacticiel, vous allez configurer une AWS Lambda fonction simple qui écoute les événements liés aux tâches Amazon ECS et les enregistre dans un flux de journal CloudWatch des journaux.

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 et External à l'aide de la console” 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 la section Événements et modèles d'événements dans le guide de l'utilisateur Amazon CloudWatch Events.

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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": [
        {
            "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,
            "memory": 300,
            "image": "httpd:2.4",
            "name": "simple-app"
```

```
        }
    ],
    "family": "console-sample-app-static"
}
```

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 panneau de navigation, choisissez Logs (Journaux) et sélectionnez le groupe de journaux pour votre fonction Lambda (par exemple, /aws/lambda/*my-fonction*).
7. Sélectionnez un flux de journaux pour afficher les données d'événement.

Envoi d'alertes Amazon Simple Notification Service en cas d'arrêt d'une tâche

Dans ce didacticiel, vous allez configurer 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	

Option	Valeur
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
Modèle d'événement	Modèle personnalisé (éditeur JSON)

Modèle d'événement	{
	"source":[
	"aws.ecs"
	],
	"detail-type":[
	"ECS Task State Change"
	],
	"detail":{

Option	Valeur

	<pre>"lastStatus":[
	"STOPPED"
	],
	"stoppedReason":[
	"Essentia
	l container in task
	exited"
	]
	}
	}</pre>

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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":"amazonlinux:2",
            "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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 des messages du journal multiligne ou de suivi de pile

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 des messages de journal divisés par l'environnement d'exécution du conteneur en suivant l'exemple suivant GitHub : FireLens Exemple : Concaténer des journaux de conteneur partiels/divisés.

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.
- Un rôle IAM d'exécution de tâche Amazon ECS.

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 l'adresse https://console.aws.amazon.com/iam/.
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 politiques, 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 de plus amples 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 "

```
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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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.

Utilisation des systèmes de fichiers Amazon EFS avec Amazon ECS à l'aide de la console

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 accéder aux données du système de fichiers dans l'ensemble de votre flotte de tâches Amazon ECS. De cette façon, vos tâches ont accès au même stockage permanent, quelle que soit l'infrastructure ou

l'instance de conteneur sur laquelle elles se retrouvent. Lorsque vous référencez votre système de fichiers Amazon EFS et votre point de montage de conteneur dans votre définition de tâche Amazon ECS, Amazon ECS prend soin de monter le système de fichiers dans votre conteneur. Les sections suivantes vous permettent de débuter à l'aide d'Amazon EFS avec Amazon ECS.

Cette fonction est prise en charge par les tâches qui utilisent à la fois les types de lancement EC2 et Fargate, mais ce didacticiel repose sur une tâche Amazon ECS qui utilise le type de lancement EC2. Ce didacticiel est également destiné à être suivi étape par étape. Toutefois, si certaines de ces ressources sont déjà créées sur votre compte, vous pouvez ignorer certaines étapes.

> **Note**
>
> Amazon EFS peut ne pas être disponible dans toutes les régions. Pour plus d'informations sur les régions prenant en charge Amazon EFS, veuillez consulter Points de terminaison et quotas Amazon Elastic File System dans la Références générales AWS (langue française non garantie).

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 plus d’informations, consultez the section called “Exemples de cluster”.

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 instances Amazon EC2 à votre cluster, développez Infrastructure, puis sélectionnez Instances Amazon EC2. 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.
     - Dans EC2 instance type (Type d'instance EC2), 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 des instances Amazon EC2 et le système de fichiers Amazon EFS

Lors de cette étape, vous créez un groupe de sécurité pour vos instances Amazon EC2 autorisant le trafic réseau entrant sur le port 80 et pour votre système de fichiers Amazon EFS autorisant l'accès entrant depuis vos instances de conteneur.

Créez un groupe de sécurité pour vos instances Amazon EC2 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, veuillez consulter Création d'un groupe de sécurité dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

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 des instances Amazon EC2 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 monterez le système de fichiers Amazon EFS sur une instance Amazon EC2 et y ajouterez 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 instance Amazon EC2 et monter le système de fichiers Amazon EFS

1. Ouvrez la console Amazon EC2 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 des instances Amazon EC2 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.

Maintenant, connectez-vous à l'instance Amazon EC2 et ajoutez du contenu au système de fichiers Amazon EFS.

Pour vous connecter à l'instance Amazon EC2 et ajouter du contenu au système de fichiers Amazon EFS

1. Envoyez une requête SSH à l'instance Amazon EC2 que vous avez créée. Pour en savoir plus, consultez Connexion à votre instance Linux dans le guide de l'utilisateur Amazon EC2 pour les instances Linux.

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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": "nginx"
        }
    ],
    "volumes": [
        {
            "name": "efs-html",
            "efsVolumeConfiguration": {
                "fileSystemId": "fs-1324abcd",
                "transitEncryption": "ENABLED"
            }
        }
    ],
    "family": "efs-tutorial",
    "executionRoleArn":"arn:aws::iam::111122223333:role/ecsTaskExecutionRole"
}
```

> **Note**
>
> Vous pouvez ajouter les autorisations suivantes à votre rôle IAM d'exécution de tâches Amazon ECS afin de permettre à l'agent Amazon ECS de localiser et de monter un système de fichiers Amazon EFS sur une tâche au démarrage.
>
> - `elasticfilesystem:ClientMount`
> - `elasticfilesystem:ClientWrite`
> - `elasticfilesystem:DescribeMountTargets`
> - `elasticfilesystem:DescribeFileSystems`

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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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.

Pour démarrer un service à partir de	Étapes
	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. Pour Launch Type (Type de lancement), choisissez 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 de systèmes de fichiers FSx for Windows File Server avec Amazon ECS

FSx for Windows File Server fournit des serveurs de fichiers Microsoft Windows entièrement gérés, qui sont appuyés par un système de fichiers Windows entièrement natif. Lorsque vous utilisez FSx for Windows File Server avec Amazon ECS, vous pouvez allouer vos tâches Windows avec un stockage de fichiers permanent, distribué, partagé et statique. Pour de plus amples informations, veuillez consulter Qu'est-ce que FSx for Windows File Server ? dans le Guide de l'utilisateur FSx for Windows File Server.

Vous pouvez utiliser FSx for Windows File Server pour déployer des applications Windows nécessitant un accès au stockage externe partagé, au stockage régional hautement disponible ou au stockage haut débit. Vous pouvez monter un ou plusieurs volumes de système de fichiers FSx for Windows File Server sur un conteneur s'exécutant sur une instance Windows EC2. Vous pouvez partager des volumes de système de fichiers FSx for Windows File Server entre plusieurs conteneurs au sein d'une seule tâche ECS.

> **Note**
>
> FSx for Windows File Server peut ne pas être disponible dans toutes les régions. Pour plus d'informations sur les régions qui prennent en charge FSx for Windows File Server, veuillez

consulter Points de terminaison et quotas Amazon FSx dans la Références générales AWS (langue française non garantie).

Dans ce didacticiel, vous lancez une instance Windows optimisée pour ECS qui héberge un système de fichiers FSx for Windows File Server et des conteneurs qui peuvent accéder au système de fichiers. Pour ce faire, vous devez d'abord créer un Microsoft Active Directory AWS Directory Service AWS géré. Ensuite, créez un système de fichiers Amazon FSx for Windows File Server et un cluster ECS avec une instance Amazon EC2 et une définition de tâche ECS. 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.

Il faut 20 à 45 minutes chaque fois que vous lancez ou supprimez le système de fichiers Active Directory ou FSx for 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. veuillez consulter Configurer l'utilisation d'Amazon ECS.
- (Facultatif) Une paire de clés `PEM` permettant de vous connecter à votre instance Windows EC2 via un accès RDP. Pour de plus amples informations sur la création d'une paire de clés, veuillez consulter Paires de clés Amazon EC2 et instances Windows dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.
- 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 répertoire Active Directory, le système de fichiers FSx for Windows File Server, le cluster ECS et l'instance EC2 doivent se trouver dans votre VPC. Pour plus d'informations sur les VPC et Active Directory, consultez Configurations de l'assistant de la console Amazon VPC et AWS Managed Microsoft AD Prerequisites.
- 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.

   - Rôle Amazon EC2 EC2 ContainerServicefor
   - Amazon SMS ManagedInstanceCore
   - Amazon SMS DirectoryServiceAccess

   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 F SxReadOnlyAccess
   - Amazon SMS ReadOnlyAccess

- AmazonECS TaskExecutionRolePolicy

Étape 2 : Créer un Active Directory (AD) Windows

1. Suivez les étapes décrites dans la section Create Your AWS Managed AD Directory Directory Guide du Guide d'administration du AWS Directory Service. Utilisez le VPC que vous avez désigné pour ce didacticiel. À l'étape 3 de Create Your AWS Managed AD Directory, enregistrez le nom d'utilisateur et le mot de passe à utiliser dans une étape ultérieure. Notez également le nom de domaine complet pour les prochaines étapes. Vous pouvez effectuer l'étape suivante pendant la création du répertoire Active Directory.
2. Créez un AWS secret du Gestionnaire de Secrets à utiliser dans les étapes suivantes. Pour plus d'informations, consultez Getting Started with AWS 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. Celles-ci sont décrites dans la section Groupes de sécurité Amazon VPC du Guide de l'utilisateur FSx for Windows File Server. 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 autorise l'envoi du trafic vers et depuis le système de fichiers FSx for 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 console EC2 et sélectionnez Security groups (Groupes de sécurité) depuis 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 instance EC2 ultérieurement dans ce didacticiel, modifiez la source RDP de la règle entrante avec l'adresse IP publique de votre instance EC2 tel que décrit dans la section Connectez-vous à votre instance Windows du Guide de l'utilisateur Amazon EC2 pour les instances Windows.

Étape 4 :Créer un système de fichiers Amazon FSx for Windows File Server

Une fois que votre groupe de sécurité est vérifié et mis à jour et que votre répertoire Active Directory est créé avec l'état active (actif), créez le système de fichiers FSx for Windows File Server dans le même VPC que votre répertoire Active Directory. Suivez les étapes suivantes pour créer un système de fichiers FSx for Windows File Server.

Créez votre premier système de fichiers.

1. Ouvrez la console Amazon FSx.
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 Select file system type (Sélectionner le type de système de fichiers), choisissez FSx for Windows File Server, puis Next (Suivant). 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 instance EC2 pendant 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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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. Étendez l'infrastructure, AWS Fargate effacez (sans serveur), puis sélectionnez les instances Amazon EC2.

    - 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.
        - Pour Type d'instance EC2, choisissez t2.medium ou t2.micro.
7. Choisissez Créer.

Étape 6 : créer une instance Amazon EC2 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 console Amazon EC2 pour lancer l'instance.

    a. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.

    b. Dans la barre de navigation, sélectionnez la région à utiliser.

    c. Sur le tableau de bord EC2, sélectionnez Launch instance (Lancer une 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. Cette option de jonction de domaines joint votre répertoire AD lorsque l'instance EC2 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 Advanced Details (Détails avancés), collez le script suivant dans le champ User data (Données utilisateur), en remplaçant *cluster_name* 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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.

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 fichiers FSx. 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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 pour votre rôle d'exécution de tâche et les détails sur votre système de fichiers FSx, 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 for Windows File Server est Available (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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 Launch Type (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 FSx for Windows File Server ou du répertoire AD prend 20 à 45 minutes. Vous devez attendre que les opérations de suppression du système de fichiers FSx for Windows File Server soient terminées avant de lancer les opérations de suppression du répertoire AD.

Supprimez le système de fichiers FSx for Windows File Server.

1. Ouvrez la console Amazon FSx.
2. Cliquez sur la case d'option située à gauche du système de fichiers FSx for 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 partir de l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans le volet de navigation, choisissez Clusters, puis choisissez fsx-windows-cluster.
3. Choisissez Supprimer le cluster.
4. Saisissez l'expression, puis choisissez Supprimer.

Résiliez l'instance EC2.

1. Ouvrez la console Amazon EC2.
2. Dans le menu de gauche, sélectionnez Instances.
3. Cochez la case située à gauche de l'instance EC2 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).

Déploiement de Fluent Bit sur des conteneurs Amazon ECS pour Windows

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 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`

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, consultez Installing the AWS Command Line Interface (Installation de).
- 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 sur lequel l'instance EC2 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 plus d'informations, consultez 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.

   ```
   {
       "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 console Amazon EC2 pour lancer l'instance.
   a. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
   b. Dans la barre de navigation, sélectionnez la région à utiliser.
   c. Sur le tableau de bord EC2, sélectionnez Launch instance (Lancer une 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 Advanced Details (Détails avancés), collez le script suivant dans le champ User data (Données utilisateur), en remplaçant *`cluster_name`* 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 :

- *`région`* 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 l'étiquette de l'enregistrement du journal par le format de sortie. *`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 instance EC2 pour empêcher tout 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`* par la région dans laquelle votre tâche s'exécute

```
{
  "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`* par la région dans laquelle votre service fonctionne

   ```
   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`* par la région dans laquelle les tâches de votre service fonctionnent

```
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* par la région dans laquelle votre tâche s'exécute

   ```
   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`* par la région dans laquelle votre tâche s'exécute

   ```
   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 plus d'informations, consultez the section called “Arrêter une tâche Amazon ECS”.
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 des GMSA pour les conteneurs Windows sur Amazon EC2

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 s'exécutant avec un compte gMSA attend de son instance Amazon EC2 hôte qu'elle récupère les informations d'identification gMSA auprès du contrôleur de domaine Active Directory et qu'elle les communique à l'instance de conteneur. Pour plus d'informations, consultez Création de groupes gMSA pour des 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 lorsqu'il s'agit d'utiliser des groupes gMSA pour des 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 plus d'informations, consultez 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.

Options de stockage de données gMSA par système d'exploitation

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
Amazon EC2 Systems Manager Parameter Store	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 plus d'informations, consultez AWS Direct Connect.
- Vous disposez d'un compte gMSA existant dans l'Active Directory. Pour plus d'informations, consultez Création de groupes gMSA pour des 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'instance Amazon EC2.
  - Si vous stockez la spécification des informations d'identification dans SSM Parameter Store, les autorisations IAM d'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 avec Domainless 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.

```
{
    "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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
                "arn:aws:ssm:region: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 de conteneurs Windows avec Domainless 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 AWS utilisateur 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.
- 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 plus d'informations, consultez 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.

Options de stockage de données gMSA par système d'exploitation

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
Amazon EC2 Systems Manager Parameter Store	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 instance EC2. 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.

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 à partir d'un serveur ou d'un client Windows ayant accès au domaine. Remplacez `ExampleAccount` par le nom que vous souhaitez attribuer à votre compte gMSA.

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
 "contoso" -ServicePrincipalNames "host/ExampleAccount", "host/contoso" -
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.

```
PS C:\> New-ADUser -Name "ExampleAccount" -AccountPassword (ConvertTo-
SecureString -AsPlainText "Test123" -Force) -Enabled 1 -PasswordNeverExpires 1
```

e.
```
PS C:\> Add-ADGroupMember -Identity "ExampleAccountHosts" -Members
 "ExampleAccount"
```

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. 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\":\"ExampleAccount\",\"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-joined un 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": "contoso",
    "DnsName": "contoso",
    "NetBiosName": "contoso"
  },
  "ActiveDirectoryConfig": {
    "GroupManagedServiceAccounts": [
      {
        "Name": "ExampleAccount",
        "Scope": "contoso"
      },
      {
        "Name": "ExampleAccount",
        "Scope": "contoso"
      }
    ],
    "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 CLI les SDK et AWS CloudFormation. 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 des SDK AWS Management Console AWS CLI, 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 des instances Amazon EC2. Chaque instance de conteneur s'enregistre dans un cluster Amazon ECS. Avant de pouvoir lancer des instances Amazon EC2 et de les enregistrer dans un cluster, vous devez créer le rôle IAM que vos instances de conteneur utiliseront.

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 `ecsTaskExecutionRole` 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`.

```
{
"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:aws-region:111122223333:secret:gmsa-plugin-input"
    }
  ]
}
```

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 IAM requises pour 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 :

```
{
    "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 :

   ```
   {
       "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 Windows Amazon EC2 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 Windows Amazon EC2 configurée pour Amazon ECS dans le AWS Management Console, consultez. Lancement 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 ID différent de celui de l'instance EC2.

```
$ 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. Résiliez l'instance Amazon EC2. Ensuite, l'instance de conteneur du cluster sera automatiquement supprimée au bout d'une heure. Pour plus d'informations, consultez Gestion de capacité.

   Vous pouvez détecter et résilier l'instance à l'aide de la console Amazon EC2. Vous pouvez également exécuter la commande suivante. Pour exécuter la commande, recherchez l'ID d'instance EC2 dans le résultat de la commande `aws ecs describe-container-instances` de l'étape 1, Étape 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, voirVérification des erreurs de tâches arrêtées.

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, veuillez consulter Résoudre des problèmes de comptes de service administré de groupe pour des conteneurs 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.

Utilisation gMSA pour les Linux conteneurs sur Amazon EC2

Amazon ECS prend en charge l'authentification Active Directory pour les conteneurs Linux sur 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 conteneur Linux qui s'exécute avec gMSA repose sur le démon `credentials-fetcher` qui s'exécute sur l'instance Amazon EC2 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.

Rubriques

- Considérations
- Prérequis
- Configuration de conteneurs Linux compatibles avec gMSA sur Amazon ECS
- Fichier de spécification des informations d'identification

Considérations

Tenez compte des points suivants avant d'utiliser des gMSA pour les conteneurs Linux :

- Si vos conteneurs fonctionnent sur EC2, vous pouvez utiliser des gMSA pour les conteneurs Windows et Linux. 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.

Options de stockage de données gMSA par système d'exploitation

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
Amazon EC2 Systems Manager Parameter Store	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 plus d'informations, consultez AWS Direct Connect.
- Vous disposez d'un compte gMSA existant dans l'Active Directory. Pour plus d'informations, consultez Utilisation gMSA pour les Linux conteneurs sur Amazon EC2.
- 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 d'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 utilisateur EC2 dans un modèle de lancement EC2 réutilisable pour installer le démon `credentials-fetcher`.

1. Décidez si vous souhaitez joindre les instances au domaine ou utiliser le gMSA sans domaine.
   - Jonction des instances EC2 au domaine Active Directory

     - Jointure des instances par les données utilisateur

       Ajoutez les étapes permettant de joindre le domaine Active Directory à vos données utilisateur EC2 dans un modèle de lancement EC2. 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'instance EC2 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 démon `credentials-fetcher` avec un script de données utilisateur dans votre modèle de lancement EC2. 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-fetch && 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-fetch && 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`.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": [
   ```

```
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:ssm:aws-region: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.

```
{
    "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 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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": [
                "arn:aws:ssm:aws-region: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 plus d’informations, consultez 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 plus d’informations, consultez 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`.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "arn:aws:secretsmanager:aws-region: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.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListObject"
            ],
            "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 `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.

Pour PowerShell, utilisez la commande suivante :

```
$ Write-S3Object -BucketName "MyBucket" -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://MyBucket/ecs-domainless-gmsa-credspec
```

Utilisation d'EC2 Image Builder pour créer des AMI personnalisées optimisées pour Amazon ECS

AWS vous recommande d'utiliser les AMI optimisées pour Amazon ECS, car elles sont préconfigurées conformément aux exigences et aux recommandations pour exécuter vos charges de travail de conteneur. 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 le mode de fonctionnement d'EC2 Image Builder dans le guide de l'utilisateur d'EC2 Image Builder.

Nous vous recommandons d'utiliser l'une des AMI optimisées pour Amazon ECS suivantes comme « image parent » dans EC2 Image Builder :

- Linux
  - AL2023 x86 optimisé pour Amazon ECS
  - 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 d'EC2 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

pour mettre à jour automatiquement vos instances EC2 dans votre cluster Amazon ECS avec les nouvelles AMI, 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 AWS du SDK. Lorsque vous spécifiez une image parent dans votre recette, vous pouvez spécifier un ID d'AMI EC2, un ARN d'image Image Builder, un identifiant de AWS Marketplace produit ou une image de conteneur. AWS publie publiquement les identifiants AMI et les ARN des images Image Builder des AMI optimisées pour Amazon ECS. 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'AMI optimisées pour Amazon ECS.

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.

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.

Rubriques

- Résolution des problèmes avec ECS Exec
- Résolution des problèmes liés à Amazon ECS Anywhere
- Vérification des erreurs de tâches arrêtées
- Codes d'erreur des tâches arrêtées
- CannotPullContainer erreurs de tâche
- Messages d'événement de service
- Valeur d'UC ou de mémoire non valide
- CannotCreateContainerError: API error (500): devmapper
- Résolution des problèmes liés aux équilibreurs de charge des services
- Résolution des problèmes de connexion à un volume Amazon EBS
- Dépannage de la scalabilité automatique de service
- Utiliser la sortie de débogage Docker
- Emplacements des fichiers journaux Amazon ECS
- Collecteur de journaux Amazon ECS
- Diagnostics d'introspection d'agent
- Diagnostics Docker
- AWS Fargate limitation des quotas
- Motifs d'échec d'API
- Bonnes pratiques pour gérer les problèmes de régulation d'Amazon ECS

Résolution des problèmes avec ECS Exec

Voici des notes de résolution pour vous aider à diagnostiquer d'où viennent les erreurs lorsque vous utilisez ECS Exec.

Vérifier à l'aide d'Amazon ECS 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 différentes API 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 plus d’informations, consultez Autorisations IAM requises pour 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.

Résolution des 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 plus d'informations, consultez Enregistrement d'une instance externe dans un cluster.
- 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 plus d'informations, consultez Rôle IAM 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 plus d'informations, consultez Annuler l'enregistrement d'une instance externe.

  > **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 avec ECS Anywhere.

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 plus d'informations, consultez Collecteur de journaux Amazon ECS.

Vérification des erreurs de tâches arrêtées

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. Vous pouvez consulter les erreurs liées aux tâches arrêtées de ce type dans la console Amazon ECS en consultant la tâche arrêtée et en l'inspectant pour détecter les messages d'erreur.

Si votre définition de tâche utilise le pilote de `awslogs` journal, les journaux d'application écrits dans Amazon CloudWatch Logs sont affichés dans l'onglet Logs de la console Amazon ECS tant que la tâche arrêtée apparaît.

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 les AWS SDK AWS Management Console AWS CLI, l'API Amazon ECS ou les outils qui utilisent les SDK et l'API. 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. Pour plus d'informations, consultez Messages d'événement de service.

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'instance Amazon EC2 hôte ou sur un ordinateur externe. Pour plus d'informations, consultez Emplacements des fichiers journaux Amazon ECS.

> **Important**
>
> Les tâches arrêtées apparaissent uniquement dans la console Amazon ECS et dans AWS les SDK pendant au moins une heure après l'arrêt de la tâche. AWS CLI Ensuite, les détails de la tâche arrêtée expirent et ne sont plus disponibles dans Amazon ECS.
> Amazon ECS envoie également des événements de modification de l'état des tâches à Amazon EventBridge. Vous ne pouvez pas afficher les événements dans EventBridge. Au lieu de cela, vous créez des règles pour envoyer les événements vers un autre stockage persistant tel qu'Amazon CloudWatch Logs. Vous pouvez utiliser l'espace de stockage pour afficher votre tâche arrêtée lorsque celle-ci a expiré dans la console Amazon ECS. Pour plus d'informations, consultez Événements de modification de l'état des tâches Amazon ECS.
> Pour un exemple de EventBridge configuration permettant d'archiver les événements Amazon ECS dans Amazon CloudWatch Logs, consultez ECS Stopped Tasks in CloudWatch Logs sur le GitHub site Web.

Procédez comme suit pour consulter les erreurs des tâches arrêtées.

Console

AWS Management Console

Les étapes suivantes peuvent être utilisées pour vérifier la présence d'erreurs dans les tâches arrêtées lors de l'utilisation du nouveau AWS Management Console.

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.

2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur la page Cluster : ***name*** (Cluster : nom), choisissez l'onglet Tasks (Tâches).
5. Configurez le filtre pour afficher les tâches arrêtées. Pour Statut souhaité du filtre, choisissez Arrêté ou Tout statut souhaité.

   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 us-west-2
```

2. Décrivez la tâche arrêtée pour récupérer la valeur `stoppedReason` dans la réponse.

```
aws ecs describe-tasks \
     --cluster cluster_name \
     --tasks arn:aws:ecs:us-west-2:account_id:task/cluster_name/task_ID \
     --region us-west-2
```

Ressources supplémentaires

Les pages suivantes fournissent des informations supplémentaires sur les codes d'erreur :

- Pourquoi ma tâche Amazon ECS est-elle arrêtée ?
- Codes d'erreur des tâches arrêtées

Codes d'erreur des tâches arrêtées

Voici les messages d'erreur possibles que vous pouvez recevoir lorsqu votre tâche Fargate est arrêtée de façon inattendue. Les messages d'erreur sont renvoyés par l'agent de conteneur. Le préfixe dépend de la version de plateforme utilisée par la tâche.

Pour vérifier la présence d'un message d'erreur dans vos tâches arrêtées à l'aide du AWS Management Console, voirVérification des erreurs de tâches arrêtées.

`ContainerRuntimeTimeoutError`, `DockerTimeoutError` (version 1.3.0 ou antérieure de la plateforme Linux)

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.

Exemple : `ContainerRuntimeTimeoutError: Could not transition to running; timed out after waiting 1m: <reason>`

`CannotStartContainerError`

Cette erreur se produit lorsqu'un conteneur ne peut pas être démarré.

Exemple : `CannotStartContainerError: failed to get container status: <reason>`

`CannotStopContainerError`

Cette erreur se produit lorsqu'un conteneur ne peut pas être arrêté.

Exemple : `CannotStopContainerError: failed sending SIGTERM to container: <reason>`

`CannotInspectContainerError`

Cette erreur se produit lorsque l'agent de conteneur ne peut pas décrire le conteneur via son environnement d'exécution.

Lorsque vous utilisez la version 1.3 ou antérieure de la plateforme, l'agent Amazon ECS renvoie la raison à partir de Docker.

Lorsque vous utilisez la version 1.4 `1.4.0` ou ultérieure de la plateforme (Linux) `1.0.0` ou ultérieure (Windows), l'agent Fargate renvoie la raison de. `containerd`

Exemple : `CannotInspectContainerError: <reason>`

`ResourceInitializationError`

Cette erreur se produit lorsque l'agent de conteneur de votre tâche Fargate ne parvient pas à créer ou à démarrer les ressources nécessaires pour démarrer le conteneur ou la tâche à laquelle il appartient.

Une cause fréquente de cette erreur est l'utilisation d'un VPC dont la résolution DNS n'est pas activée.

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).

Exemple : `ResourceInitializationError: failed to initialize logging driver: <reason>`

`CannotPullContainerError`

Cette erreur se produit lorsque l'agent ne parvient pas à extraire l'image de conteneur spécifiée dans la définition de tâche. Pour plus d'informations, consultez CannotPullContainer erreurs de tâche.

Exemple : `CannotPullContainerError: <reason>`

`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).

Exemple : `CannotCreateVolumeError: <reason>`

`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).

Exemple : `ContainerRuntimeError: failed to create container IO set: <reason>`

`OutOfMemoryError`

Cette erreur se produit lorsqu'un conteneur se ferme en raison de processus dans le conteneur consommant plus de mémoire que ce qui a été alloué dans la définition de tâche.

Exemple : `OutOfMemoryError: container killed due to memory usage`

`InternalError`

Cette erreur se produit 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.

Exemple : `InternalError: <reason>`

`TaskFailedToStart`

Cette erreur se produit lorsqu'une pièce jointe ENI est demandée. Amazon EC2 gère de manière asynchrone le provisionnement de l'ENI. 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.

Exemple : `InternalError: <reason>`

`SpotInterruption`

Cette erreur se produit lorsqu'il n'y a pas de capacité Fargate Spot ou lorsque Fargate reprend la capacité de Fargate Spot.

Exemple : `SpotInterruption: Your Spot Task was interrupted`

Exemple : `InternalError: <reason>`

`TaskFailedToStart`

Cette erreur se produit lorsque le sous-réseau qui héberge les instances ne possède pas suffisamment d'adresses IP. Le nombre d'adresses IP disponibles est disponible sur la page

détaillée du sous-réseau ou en utilisant `describe-subnets`. Pour plus d'informations, consultez View your subnet (Afficher votre sous-réseau dans le Guide de l'utilisateur Amazon VPC.

Exemple : `Unexpected EC2 error while attempting to Create Network Interface with public IP assignment enabled in subnet 'subnet-id': InsufficientFreeAddressesInSubnet`

`TaskFailedToStart`

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 plus d'informations, consultez Types de lancement Amazon ECS.

Exemple : `The selected task definition is not compatible with the selected compute strategy`

CannotPullContainer erreurs de tâche

L'erreur suivante indique que lorsque vous créez une tâche, l'image de conteneur spécifiée n'a pas pu être récupérée.

> **Note**
>
> La version 1.4 de la plateforme Fargate tronque les messages d'erreur longs.

Connection timed out (Expiration de la connexion)

Lors du lancement d'une tâche Fargate, son interface réseau Elastic a besoin d'un acheminement vers Internet pour extraire les images de conteneur. Si vous recevez une erreur similaire à celle ci-dessous lors du lancement d'une tâche, cela signifie qu'il n'existe pas d'itinéraire vers Internet :

```
CannotPullContainerError: API error (500): Get https://111122223333.dkr.ecr.us-east-1.amazonaws.com/v2/: net/http: request canceled while waiting for connection"
```

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 plus d'informations, consultez Exécuter 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.

Contexte 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.

Image non trouvée (Image introuvable)

Lorsque vous spécifiez une image Amazon ECR dans votre définition de conteneur, vous devez utiliser l'URI complet de votre référentiel Amazon ECR ainsi que le nom de l'image dans ce référentiel. Si le référentiel ou l'image est introuvable, vous recevez l'erreur suivante :

```
CannotPullContainerError: API error (404): repository 111122223333.dkr.ecr.us-
east-1.amazonaws.com/<repo>/<image> not found
```

Pour résoudre ce problème, vérifiez l'URI du référentiel et le nom de l'image. Assurez-vous également que vous avez configuré l'accès approprié à l'aide du rôle IAM d'exécution de tâche. Pour plus d'informations sur le rôle d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Problème de connexion au point de terminaison Amazon ECR

Si vous essayez d'extraire une image Amazon ECR et que vous ne disposez pas des autorisations appropriées pour le point de terminaison Amazon ECR, une erreur similaire à la suivante s'affiche :

```
CannotPullContainerError: API error
```

Pour résoudre ce problème, Amazon ECS doit communiquer avec le point de terminaison Amazon ECR. Pour plus d'informations sur la manière de résoudre ces problèmes, consultez l'article Comment puis-je résoudre l'erreur Amazon ECR « CannotPullContainerError : erreur d'API » dans Amazon ECS sur le AWS Support site Web.

Insufficient disk space (Espace disque insuffisant)

Si le volume racine de votre instance de conteneur ne contient pas assez d'espace disque lors de l'extraction de l'image de conteneur, un message d'erreur similaire à ce qui suit s'affiche :

```
CannotPullContainerError: write /var/lib/docker/tmp/GetImageBlob111111111: no space
 left on device
```

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, comme dans l'exemple précédent, 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 Configurer des pilotes de journalisation 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 plus d'informations, consultez Utilisation du pilote du journal awslogs.

Limite de débit 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 :

```
ERROR: toomanyrequests: Too Many Requests.
```

```
You have reached your pull rate limit. You may increase the limit by authenticating
 and upgrading: https://www.docker.com/increase-rate-limits.
```

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 Docker Hub et que vous devez authentifier vos extractions Docker pour vos instances de conteneur, consultez Private registry authentication for container instances dans le Guide du développeur Amazon Elastic Container Service.

Échec de copie d'image

Si vous recevez une erreur similaire à celle ci-dessous lors du lancement d'une tâche, cela signifie qu'il n'existe pas d'accès à l'image :

```
CannotPullContainerError: ref pull has been retried 1 time(s): failed to copy:
 httpReaderSeeker: failed open: unexpected status code
```

Pour résoudre ce problème, vous pouvez :

- 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 en extraction refusé

Si vous recevez une erreur similaire à celle ci-dessous lors du lancement d'une tâche, cela signifie qu'il n'existe pas d'accès à l'image :

```
CannotPullContainerError: pull access denied
```

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.

Adresse mémoire non valide ou déréférencement de pointeur nul

Si vous recevez une erreur similaire à celle ci-dessous lors du lancement d'une tâche, cela signifie qu'il n'existe pas d'accès à l'image :

```
CannotPullContainerError: containerd: pull command failed: panic: runtime error:
 invalid memory address or nil pointer dereference
```

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 d'extraction de la configuration de l'image

Si vous recevez une erreur similaire à celle ci-dessous lors du lancement d'une tâche, cela signifie qu'il existe une limite de débit atteinte ou une erreur réseau :

```
CannotPullContainerError: error pulling image conf/error pulling image configuration
```

Pour résoudre ce problème, consultez Comment puis-je résoudre l'erreur « CannotPullContainerError » dans ma tâche de type de lancement Amazon ECS EC2.

Pour plus d'informations sur les erreurs de tâches arrêtées, consultez Codes d'erreur pour les tâches arrêtées dans le Guide de l'utilisateur Amazon Elastic Container Service pour AWS Fargate.

Messages d'événement de service

Lorsque vous voulez résoudre un problème lié à un service, le premier endroit à consulter pour obtenir des informations est 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 le préfixe service (***service-name***). Pour vous assurer que cette vue d'événement est utile, nous ne montrons que les `100` événements les plus récents. Les messages d'événements dupliqués sont omis jusqu'à ce que la cause soit résolue ou au bout de six heures. 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. Le préfixe de ces événements est Message. Les `10` événements de mise à l'échelle les plus récents sont affichés. Ces événements ne peuvent se produire que lorsqu'un service est configuré avec une stratégie de mise à l'échelle Application Auto Scaling.

Procédez comme suit pour afficher vos messages d'événement de service actuels.

Console

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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 *service-name du* cluster *par défaut*, qui fournira les derniers messages d'événements de service.

```
aws ecs describe-services \
    --cluster default \
    --services service-name \
    --region us-west-2
```

Messages d'événement de service

Voici quelques exemples de messages d'événement de service que vous pouvez rencontrer dans la console Amazon ECS.

le service ***nom-service*** 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.

le service (***nom-service***) 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 :

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 plus d’informations, consultez 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 Gestion de la mémoire des instances de conteneurs.

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'ENI qui peuvent être attachées aux instances Amazon EC2 est limité. Il n'y a pas d'instances de conteneur dans le cluster qui présente 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 de plus amples informations, veuillez consulter Adresses IP par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.
- 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 de plus amples informations, veuillez consulter Adresses IP par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.
- Si vous avez opté pour le paramètre de compte `awsvpcTrunking` et que vous avez lancé de nouvelles instances de conteneur utilisant un type d'instance pris en charge après ce choix, des ENI supplémentaires sont disponibles. Pour plus d'informations, consultez Types d'instances Amazon EC2 pris en charge.

Pour en savoir plus sur l'acceptation du paramètre de compte `awsvpcTrunking`, consultez Jonction d'interface réseau Elastic.

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 lancement EC2. 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.

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 de tâche et Configuration de l'agent de conteneur Amazon ECS.

le service (***nom-service***) 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.

le service (***nom-service***) 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 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***) (instance ***instance-id***) n'est pas fonctionnel dans (elb ***elb-name***) en raison de (raison pour laquelle l'instance a échoué à au moins le UnhealthyThreshold 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. Pour plus d'informations, consultez Résolution des problèmes liés aux équilibreurs de charge des services.

le service (***nom-service***) ne parvient pas à démarrer les tâches de manière cohérente.

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 plus d'informations, consultez 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 service (***nom-service***) 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.

Le service (***nom-service***) n'a pas pu arrêter ou démarrer des tâches lors d'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 maximum 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.

le service (***`nom-service`***) n'a pas pu placer une 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 plus d'informations, consultez Service Quotas. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

le service (***`nom-service`***) n'a pas pu placer une tâche. Motif : Erreur interne.

Les éléments suivants présentent les causes possibles de cette erreur :

- 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.
- Vous essayez d'exécuter une définition de tâche qui utilise l'architecture ARM sur Fargate Spot.

le service (***nom-service***) n'a pas pu placer une 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 plus d'informations, consultez Service Quotas. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

le service (***nom-service***) n'a pas pu placer une 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 plus d'informations, consultez Service Quotas. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

le service (***nom-service***) n'a pas pu placer une tâche. Motif : Vous avez atteint la limite du nombre des vCPU que vous pouvez exécuter simultanément

AWS Fargate est en train de passer de quotas basés sur le nombre de tâches à des quotas basés sur le vCPU.

Vous pouvez demander une augmentation de quota pour le quota basé sur le VCPU Fargate. Pour plus d'informations, consultez Service Quotas. Pour demander une augmentation de quota Fargate, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

le service (***nom du service***) n'a pas pu atteindre l'état stable car l'ensemble de tâches (***ID de l'ensemble de tâches***) n'a pas pu effectuer une mise à l'échelle horizontale. Raison : le nombre de tâches souhaité est supérieur au nombre de tâches protégées.

Le service a plus de tâches protégées que le nombre souhaité de 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.

le service (***service-name***) n'a pas pu atteindre un état stable. 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 :

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 plus d'informations, consultez 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 Gestion de la mémoire des instances de conteneurs.

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'ENI pouvant être attachés aux instances Amazon EC2 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 de plus amples informations, veuillez consulter Adresses IP par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.
- 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 de plus amples informations, veuillez consulter Adresses IP par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.
- Si vous avez opté pour le paramètre de compte `awsvpcTrunking` et que vous avez lancé de nouvelles instances de conteneur utilisant un type d'instance pris en charge après ce choix, des ENI supplémentaires sont disponibles. Pour plus d'informations, consultez Types d'instances Amazon EC2 pris en charge.

Pour en savoir plus sur l'acceptation du paramètre de compte `awsvpcTrunking`, consultez Jonction d'interface réseau Elastic.

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 lancement EC2 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.

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 de tâche et Configuration de l'agent de conteneur Amazon ECS.

le service (***`nom-service`***) n'a pas pu placer une 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 Fargate ou les instances de conteneur EC2 soient disponibles.
- Relancez le service et spécifiez des sous-réseaux supplémentaires.

le déploiement du ***`service (nom du service`***) a échoué : 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. Codes d'erreur des tâches arrêtées

service (***`nom du service) Expiration`*** du délai d'attente du démarrage de l'agent Amazon ECS. Veuillez consulter les journaux sur « /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 plus d’informations, consultez 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
```

***l'ensemble de tâches du service (service-name) (TaskSet-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. Vous devez supprimer et recréer le service. Ne supprimez aucun groupe cible Elastic Load Balancing sauf si le service Amazon ECS correspondant est déjà supprimé.

***l'ensemble de tâches du service (service-name) (TaskSet-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.

Valeur d'UC ou de mémoire non valide

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. For more information, see the Troubleshooting section
 of the Amazon ECS Developer Guide.
```

> **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 de processeur ou en vCPU 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 uniquement `EC2` pour le paramètre `requiresCompatibilities`, les valeurs de CPU prises en charge sont comprises entre `256` unités CPU (`0.25` vCPUs) et `16384` unités CPU (16 vCPUs). La valeur de mémoire doit être un entier, et la limite dépend de la quantité de mémoire disponible sur l'instance Amazon EC2 sous-jacente que vous utilisez.

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
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)	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
Note Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure		
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 de processeur des tâches prises en charge sont comprises entre 0,25 vCPU et 192 vCPU.

> **Note**
>
> Les paramètres d'UC et de mémoire de niveau tâche sont ignorés pour les conteneurs Windows.

CannotCreateContainerError: API error (500): devmapper

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, les AMI optimisées pour Amazon ECS à partir de la version `2015.09.d` sont lancées avec un volume de 8 Gio pour le système d'exploitation attaché à l'emplacement `/dev/xvda` et montées 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 AMI optimisée pour Amazon ECS.

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 plus d'informations, consultez 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/"
```

Résolution des problèmes liés aux équilibreurs de charge des services

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 plus d'informations, consultez 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

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 ***nom du service*** : le nom du conteneur ou le port 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 de connexion à un volume Amazon EBS

Les volumes Amazon EBS peuvent être configurés pour être joints à une tâche Amazon ECS autonome lorsque vous exécutez la tâche. Ils peuvent également être configurés pour être joints à des tâches lancées via un service Amazon ECS lorsque vous créez ou mettez à jour le service. Il est possible de joindre au maximum un volume par tâche. Si votre tâche ou votre service ne démarre pas comme prévu, vous pouvez vérifier les raisons de l'échec de l'attachement au volume à l'aide du AWS Management Console. Vous pouvez ensuite utiliser la liste des scénarios de défaillance pour les étapes suivantes. Pour plus d'informations sur les volumes Amazon EBS pour les tâches Amazon ECS, consultez la section Amazon EBS volumes.

Vérification des causes de défaillance de la connexion au volume

Pour consulter l'état de la pièce jointe d'un volume et la raison de l'échec

1. Ouvrez la console à partir de l'adresse https://console.aws.amazon.com/ecs/v2.
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. Pour le statut souhaité du filtre, choisissez Arrêté, puis choisissez la tâche arrêtée que vous souhaitez résoudre.
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.
6. Choisissez l'état de la pièce jointe pour voir la raison de l'échec apparaître dans une fenêtre contextuelle.

Pour mieux comprendre pourquoi l'attachement du volume a échoué, vous pouvez configurer Amazon EventBridge pour qu'il envoie des événements à une cible, telle que CloudWatch des groupes Amazon. Vous pouvez envoyer des événements de volume Amazon EBS et des événements de modification de l'état des tâches Amazon ECS et utiliser ces événements pour diagnostiquer les problèmes. Pour plus d'informations, voir Comment créer un groupe de CloudWatch journaux à utiliser comme cible pour une EventBridge règle ? sur AWS Re:post. Pour plus d'informations sur les événements de changement d'état de tâche Amazon ECS, consultez Événements de changement d'état de tâche. Pour plus d'informations sur les événements de volume Amazon EBS, consultez EventBridge Amazon EBS dans le guide de l'utilisateur Amazon EC2.

Scénarios d'échec de l'attachement à un volume Amazon EBS

Utilisez les scénarios suivants pour vous aider à diagnostiquer et à résoudre les problèmes que vous pourriez rencontrer lorsque vous configurez des volumes Amazon EBS pour les associer à des tâches Amazon ECS.

Rôle dans l'infrastructure ECS AWS Identity and Access Management (IAM)

Les scénarios de rôle IAM suivants peuvent entraîner des problèmes d'attachement aux volumes Amazon EBS :

- Vous ne fournissez pas de rôle IAM Amazon Resource Name (ARN). Pour associer un volume Amazon EBS à une tâche Amazon ECS, vous devez fournir un ARN pour un rôle IAM avec les autorisations nécessaires pour qu'Amazon ECS puisse gérer le volume Amazon EBS en votre nom.
- 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. Le message d'erreur suivant s'affiche : ECS n'a pas pu assumer le rôle d'infrastructure ECS configuré

« arn:aws:iam*111122223333*:role/*ecsInfrastructureRole* ». Vérifiez que le rôle transmis possède une relation de confiance appropriée avec Amazon ECS. 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.

> **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, vous avez peut-être mal configuré la politique de confiance pour le rôle.

- 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 plus d'informations, consultez 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.

AWS Key Management Service (AWS KMS) clé

Les AWS KMS principaux scénarios suivants peuvent entraîner des problèmes d'attachement aux volumes Amazon EBS :

- Vous spécifiez un ARN, un ID ou un alias de clé KMS qui n'est pas 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 Chiffrement Amazon EBS dans le guide de l'utilisateur Amazon EC2.
- 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.

Dans ces scénarios, le message d'état et de raison suivant peut s'afficher dans la vue Volumes de la tâche arrêtée AWS Management Console : IdempotentParameterMismatch« ; « Le jeton client que vous avez fourni est associé à une ressource déjà supprimée. Veuillez utiliser un autre jeton client. »

`configuredAtLaunch`paramètre de définition de tâche

Vous ne pouvez pas associer un volume Amazon EBS à une tâche si vous ne le spécifiez pas `configuredAtLaunch` dans votre définition de tâche, ou si vous le définissez sur`configuredAtLaunch`. `false` Pour éviter ce problème, définissez ce `configuredAtLaunch` paramètre `true` dans votre définition de tâche.

Nom du volume

Les scénarios de nom de volume suivants peuvent entraîner des problèmes de connexion aux volumes Amazon EBS :

- Vous ne fournissez pas de nom de volume lors de la configuration du volume.
- Le nom du volume que vous fournissez lors de la configuration ne correspond pas au nom du volume que vous spécifiez dans la définition de la tâche.

Format du système de fichiers

Les scénarios de format de système de fichiers suivants peuvent entraîner des problèmes d'attachement aux volumes Amazon EBS :

- 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. Si le type de système de fichiers ne correspond pas, la tâche ne démarre pas avec le statut suivant dans la vue Volumes de la tâche arrêtée dans le AWS Management Console : ECS a expiré lors de la configuration du volume EBS attaché à votre tâche. Pour mieux comprendre le problème, vous pouvez consulter les journaux des agents de conteneurs Amazon ECS. Pour plus d'informations, consultez les sections Emplacements des fichiers journaux Amazon ECS et Amazon ECS Log Collector.

Balises

Les scénarios de balisage suivants peuvent empêcher le lancement de la tâche Amazon ECS :

- Vous essayez de propager des balises d'un service vers un volume configuré pour être rattaché à une tâche autonome. Pour éviter ce problème, définissez cette `propagateTags` option `TASK_DEFINITION` si vous configurez un volume pour le rattacher à une tâche autonome.

- Vous essayez de fournir un type de ressource de balise non valide. Le seul type de ressource de balise pris en charge pour les volumes Amazon EBS associés à des tâches est`volume`.
- Vous essayez de fournir une valeur `propagateTags` qui n'est pas valide. Les valeurs valides pour `propagateTags` sont `TASK_DEFINITIONSERVICE`, et`NONE`.
- Vous fournissez un tag qui n'est pas valide. Pour plus d'informations, consultez la section Marquage de vos ressources Amazon ECS.
- Vous fournissez trop de balises. Amazon ECS ajoute automatiquement `AmazonECSManaged` les `AmazonECSCreated` balises et au volume. Cela signifie que 48 balises supplémentaires définies et propagées par l'utilisateur peuvent être ajoutées à un volume, pour un total de 50 balises maximum par volume.
- Vous fournissez une balise réservée déjà utilisée par Amazon ECS, comme `AmazonECSManaged` ou`AmazonECSCreated`.

Limites de volume

Si vous essayez de configurer un volume Amazon EBS pour le joindre à une tâche Amazon ECS de type lancement EC2, et si les instances de conteneur Nitro basées associées au cluster ont atteint la limite de volume, vous ne pouvez pas exécuter la tâche. Un seul volume Amazon EBS peut être attaché par tâche. La limite de volume de votre instance de conteneur détermine le nombre de tâches dont les volumes Amazon EBS sont configurés pour être jointes et qui peuvent être lancées sur une instance. Pour plus d'informations sur les limites de volume, consultez la section Limites de volume des instances dans le guide de l'utilisateur Amazon EC2.

Type d'instance de conteneur

Vous ne pouvez pas associer un volume Amazon EBS à une tâche de type lancement EC2 lancée sur une instance de conteneur basée sur le Xen système. Le message d'erreur suivant s'affiche : le service (nom du service) n'a pas pu placer de tâche car aucune instance de conteneur ne répondait à toutes ses exigences. Il manque un attribut requis par votre tâche à la correspondance la plus proche (container-instance instance-id).

Type de volume Amazon EBS

Vous ne pouvez pas associer un volume Amazon EBS à une tâche si vous configurez un volume Amazon EBS magnétique (`standard`) pour le rattacher à une tâche Fargate.

Débit d'Amazon EBS et `IOPS`

Vous ne pourrez pas associer un volume Amazon EBS à une tâche si vous configurez le débit ou `IOPS` pour un type de volume qui ne prend pas en charge ces paramètres. Vous pouvez

configurer `IOPS` uniquement pour les volumes SSD IOPS provisionnés (`io1etio2`) et les volumes SSD à usage général (`gp2etgp3`). Vous pouvez configurer le débit uniquement pour les volumes HDD à débit optimisé (`st1`), Cold HDD (`sc1`) et SSD à usage général (`gp3`).

Points de montage

Pour utiliser des volumes de données avec vos tâches Amazon ECS, vous devez configurer des points de montage dans votre définition de tâche. Pour éviter les échecs, assurez-vous de spécifier un point de montage dans votre définition de tâche et de vous référer au volume source indiqué dans la définition de tâche lors de la configuration du volume.

Régions AWS et zones de disponibilité

Vous ne pouvez pas associer de volumes Amazon EBS à des tâches si vous essayez de configurer un volume Amazon EBS lors d'un déploiement dans une Région AWS zone de disponibilité qui ne prend pas en charge l'attachement de volumes Amazon EBS. Pour obtenir Région AWS des informations sur les zones de disponibilité, consultez Régions AWS la section Zones de disponibilité pour les volumes Amazon EBS.

AMI pour les tâches Amazon ECS sur Amazon EC2

Si les instances de conteneur hôte pour vos tâches Amazon ECS dans votre cluster ne disposent pas d'une AMI optimisée pour Amazon ECS, ou si elles ont une AMI optimisée pour Amazon ECS qui ne prend pas en charge l'attachement de volume Amazon EBS, l'attachement de volume échoue. L'attachement de volume Amazon EBS est pris en charge sur les AMI `20231219` optimisées pour Amazon ECS et versions ultérieures.

Agent de conteneur

Vous ne pourrez pas associer un volume Amazon EBS à une tâche Amazon ECS de type lancement EC2 si l'agent de conteneur exécuté sur l'instance de conteneur ne l'est pas `1.79.0` ou est ultérieur. Le message d'erreur suivant s'affiche : le service (nom du service) n'a pas pu placer de tâche car aucune instance de conteneur ne répondait à toutes ses exigences. Il manque un attribut requis par votre tâche à la correspondance la plus proche (container-instance instance-id).

Version de la plateforme Fargate

Vous ne pourrez pas associer un volume Amazon EBS à une tâche Amazon ECS de type lancement Fargate si la version de la plateforme Linux Fargate ne l'est pas ou si elle est ultérieure. `1.4.0`

Défaillances de service et de tâches

Il se peut que vous rencontriez des défaillances de service ou de tâche non liées à l'attachement au volume et susceptibles d'affecter l'attachement au volume. Pour plus d'informations sur la résolution des problèmes liés aux services, consultez la section Messages relatifs aux événements du service. Pour plus d'informations sur la résolution des problèmes liés aux tâches, consultez la section Codes d'erreur des tâches arrêtées. Pour les raisons de défaillance de l'API, consultez la section Motifs de défaillance de l'API.

Dépannage de la scalabilité automatique de service

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.

Utiliser la sortie de débogage Docker

Si vous rencontrez des problèmes avec les conteneurs ou les images Docker, vous pouvez activer le mode de débogage sur votre daemon Docker. L'activation du débogage permet d'obtenir des résultats plus détaillés de la part du démon, et vous pouvez utiliser ces informations pour en savoir plus sur les raisons des problèmes rencontrés par vos conteneurs ou vos images.

L'activation de ce mode de débogage Docker peut être particulièrement utile lors de la récupération des messages d'erreur envoyés à partir des registres de conteneur, tel qu'Amazon ECR. Dans de nombreux cas, il peut constituer le seul moyen de voir ces messages d'erreur.

> **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 activer le démon Docker en mode débogage 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\"}}"
```

Emplacements des fichiers journaux 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 plus d'informations, consultez Collecteur de journaux Amazon ECS.

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 plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

`ECS_LOGFILE`

Exemples de valeur : `/ecs-agent.log`

Valeur par défaut sur Linux : Null

Valeur par défaut sur Windows : Null

Emplacement où les journaux de l'agent doivent être écrits. Si vous exécutez l'agent via`ecs-init`, méthode par défaut lors de l'utilisation de l'AMI optimisée pour Amazon ECS, le chemin intégré au conteneur est `/log` et il est `ecs-init` monté sur l'hôte. `/var/log/ecs/`

`ECS_LOGLEVEL`

Exemples de valeur : `crit`, `error`, `warn`, `info`, `debug`

Valeur par défaut sur Linux : `info`

Valeur par défaut sur Windows : `info`

Niveau de détail à consigner.

`ECS_LOGLEVEL_ON_INSTANCE`

Exemples de valeur : `none`, `crit`, `error`, `warn`, `info`, `debug`

Valeur par défaut sur Linux : `none`, si `ECS_LOG_DRIVER` est explicitement défini sur une valeur non vide ; sinon, la même valeur que `ECS_LOGLEVEL`

Valeur par défaut sur Windows : `none`, si `ECS_LOG_DRIVER` est explicitement défini sur une valeur non vide ; sinon, la même valeur que `ECS_LOGLEVEL`

Peut être utilisé pour remplacer `ECS_LOGLEVEL` et définir un niveau de détail qui doit être enregistré dans le fichier journal sur instance, distinct du niveau qui est enregistré dans le pilote de journalisation. Si un pilote de journalisation est explicitement défini, les journaux sur instance sont désactivés par défaut. Ils peuvent être réactivés avec cette variable.

`ECS_LOG_DRIVER`

Exemples de valeur : `awslogs`, `fluentd`, `gelf`, `json-file`, `journald`, `logentries` `syslog`, `splunk`

Valeur par défaut sur Linux : `json-file`

Valeur par défaut sur Windows : Non applicable

Détermine le pilote de journalisation utilisé par le conteneur de l'agent.

`ECS_LOG_ROLLOVER_TYPE`

Exemples de valeur : `size`, `hourly`

Valeur par défaut sur Linux : `hourly`

Valeur par défaut sur Windows : `hourly`

Détermine si le fichier journal de l'agent de conteneur est pivoté toutes les heures ou en fonction de sa taille. Par défaut, le fichier journal de l'agent fait l'objet d'une rotation toutes les heures.

`ECS_LOG_OUTPUT_FORMAT`

Exemples de valeur : `logfmt`, `json`

Valeur par défaut sur Linux : `logfmt`

Valeur par défaut sur Windows : `logfmt`

Détermine le format de sortie du journal. Lorsque le `json` format est utilisé, chaque ligne du journal est une carte JSON structurée.

`ECS_LOG_MAX_FILE_SIZE_MB`

Exemples de valeur : `10`

Valeur par défaut sur Linux : `10`

Valeur par défaut sur Windows : `10`

Lorsque la `ECS_LOG_ROLLOVER_TYPE` variable est définie sur`size`, elle détermine la taille maximale (en Mo) du fichier journal avant sa rotation. Si le type de substitution est défini sur `hourly`, cette variable est ignorée.

`ECS_LOG_MAX_ROLL_COUNT`

Exemples de valeur : `24`

Valeur par défaut sur Linux : `24`

Valeur par défaut sur Windows : `24`

Détermine le nombre de fichiers journaux ayant fait l'objet d'une rotation à conserver. Les fichiers journaux plus anciens sont supprimés une fois cette limite atteinte.

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"
```

Journal **`ecs-init`** Amazon ECS

Le processus `ecs-init` stocke les journaux à l'emplacement `/var/log/ecs/ecs-init.log`.

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
```

Journal d'audit des informations d'identification des rôles IAM pour les tâches

Si le fournisseur d'informations d'identification pour le rôle IAM est utilisé pour fournir des informations d'identification aux tâches, ces requêtes 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 plus d'informations, consultez 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

Un exemple d'entrée de journal figure ci-dessous.

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
```

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 plus d'informations, consultez Drainage des instances de conteneur.

```
[ec2-user ~]$ sudo bash ./ecs-logs-collector.sh
```

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 Connexion à votre instance Windows à l'aide de RDP dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.
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 plus d’informations, consultez Drainage des instances de conteneur.

```
.\ecs-logs-collector.ps1
```

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.

Diagnostics d'introspection d'agent

L'API d'introspection d'agent Amazon ECS peut fournir des informations de diagnostic utiles. Par exemple, 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 plus d'informations, consultez 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 plus d'informations, consultez Introspection des conteneurs Amazon ECS.

Diagnostics Docker

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 rubrique Docker Command Line de 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.

Dresser une liste des conteneurs Docker

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

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 conteneur *dc7240fe892a* et acheminés via la commande head 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 plus d'informations, consultez Utilisation du pilote du journal awslogs.

```
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"
```

Inspecter les conteneurs Docker

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",
...
```

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

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 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.

Limitation de l'API **RunTask**

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 de taux

Vous pouvez demander une augmentation des quotas de limitation de taux pour votre compte AWS . Pour demander un ajustement de quota, contactezAWS Support Center.

Motifs d'échec d'API

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.

Action d’API	Motif de l'échec ou de l'arrêt	Cause
`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.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
`DescribeTasks`	`TaskFailedToStart: RESOURCE:*`	En cas d'erreur `RESOURCE: CPU` , le nombre de processeurs demandé par la tâche n'est pas disponible 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 du processeur des instances Amazon EC2 définies dans le groupe Auto Scaling mappé au fournisseur de capacité. Vous devez vérifier la configuration de votre fournisseur de capacité. Pour plus d'informations sur la façon d'ajouter, d'afficher et de modifier vos fournisseurs de capacité, veuillez consulter the section called "Fournisseurs 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 instances Amazon EC2

Action d'API	Motif de l'échec ou de l'arrêt	Cause
		définies dans le groupe Auto Scaling mappé au fournisseur de capacité. Vous devez vérifier la configuration de votre fournisseur de capacité. Pour plus d'informations sur la façon d'ajouter, d'afficher et de modifier vos fournisseurs de capacité, veuillez consulter the section called "Fournisseurs de capacité".
	`TaskFailedToStart: 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 ?.
	`TaskFailedToStart: MemberOf placement constraint unsatisfied`	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 de tâche 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 façon d'ajouter, d'afficher et de modifier vos fournisseurs de capacité, veuillez consulter the section called “Fournisseurs de capacité”. Pour plus d'informations sur la gestion de vos groupes Auto Scaling, veuillez consulter Groupes Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling (langue française non garantie).

Action d'API	Motif de l'échec ou de l'arrêt	Cause
	`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 fournisseurs de capacité, veuillez consulter Fournisseurs de capacité Amazon ECS. Pour plus d'informations sur la gestion de vos groupes Auto Scaling, veuillez consulter Groupes Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling (langue française non garantie).
`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.
	`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 qui peuvent être attachées à des instances Amazon EC2 est limité et l'interface réseau principale est considérée comme l'une d'elles. Pour en savoir plus sur le nombre d'interfaces réseau prises en charge par type d'instance, consultez la section Adresses IP par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

Action d’API	Motif de l'échec ou de l'arrêt	Cause
		Les erreurs `RESOURCE:GPU` indiquent que le nombre de GPU demandés par la tâche n'est pas disponible. Vous devrez peut-être ajouter des instances de conteneur GPU à votre cluster. Pour plus d’informations, consultez Utilisation des GPU sur Amazon ECS.
	`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 ?.
	`LOCATION`	L'instance de conteneur sur laquelle vous avez tenté de lancer une tâche se trouve dans une zone de disponibi lité 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 de tâche 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'instanc e de conteneur sur laquelle vous avez essayé de lancer une tâche a été précédemm ent annulé avec Amazon ECS et cette instance ne peut pas être utilisée.
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 protectio n des tâches sur les tâches existantes ou attendez que la protection des tâches expire.
	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.

Bonnes pratiques pour 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 de régulation se produit généralement lorsque vous appelez les API Amazon ECS 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 déjà 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 appeler des API en votre nom pour provisionner des ressources. Il est important de savoir quelles AWS API Amazon ECS 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'API Amazon EC2 telles que des opérations d'API Elastic Load Balancing `CreateNetworkInterface` telles que `DescribeTargetHealth` ou pour enregistrer l'interface réseau élastique et l'équilibreur de charge, respectivement. `RegisterTarget` 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.

Surveillance de l'étranglement

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, consultez la section 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 simples telles que la suivante pour examiner `ThrottlingException` les erreurs.

```
select eventname, errorcode,eventsource,awsregion, useragent,COUNT(*) count
FROM cloudtrail-table-name
where errorcode = 'ThrottlingException'
AND eventtime between '2022-01-14T03:00:08Z' and '2022-01-23T07:15:08Z'
group by errorcode, awsregion, eventsource, username, 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 plus d'informations, consultez 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 plus d'informations, consultez 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).

Utilisation CloudWatch 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 démarrent 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 AWS Support pour augmenter les limites de régulation pertinentes et également recevoir des conseils pour les besoins spécifiques de votre application.

Références de paramètres et modèles de ressources

Les sections suivantes décrivent les paramètres de définition de tâche et de définition de service.

Rubriques

- Paramètres de définition de tâche
- Modèle de définition de tâche
- Paramètres de définition de service

Paramètres de définition de tâche

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. Pour plus d'informations, consultez the section called “Exemples de définitions de tâches liées aux options de journalisation”.

Voici des descriptions plus détaillées de chaque paramètre de définition de tâche.

Famille

`family`

Type : chaîne

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 plus d'informations, consultez 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` | `FARGATE` | `EXTERNAL`

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 fournir un rôle de tâche pour un rôle IAM qui permet aux conteneurs dans l'autorisation de tâche d'appeler en votre nom les API AWS qui sont spécifiées dans les stratégies qui lui associées. Pour plus d'informations, consultez Rôle IAM de tâche.

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 plus d'informations, consultez Configuration supplémentaire des rôles IAM pour les tâches Windows.

Rôle d'exécution de tâche

`executionRoleArn`

Type : chaîne

Obligatoire : Conditionnelle

Le 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 plus d’informations, consultez 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 Linux Amazon EC2, 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 Windows Amazon EC2, les valeurs valides sont `default` et `awsvpc`. Si aucun mode réseau n'est spécifié, le mode réseau utilisé est `default`. Pour les tâches Amazon ECS hébergées sur Fargate, `awsvpc` le mode réseau est requis.

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 `bridge`, la tâche utilise le réseau virtuel intégré de Docker sous Linux, qui s'exécute à l'intérieur de chaque instance Amazon EC2 hébergeant la tâche. Le réseau virtuel intégré sous Linux utilise le pilote réseau Docker `bridge`.

Si le mode réseau est `host`, la tâche utilise le réseau de l'hôte qui contourne le réseau virtuel intégré de Docker en mappant les ports de conteneur directement à l'ENI de l'instance Amazon EC2 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. Dans ce mode, vous ne pouvez pas exécuter plusieurs tâches de la même définition de tâche sur une instance Amazon EC2 unique.

> **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.

Pour les types de lancement Amazon EC2, si le mode réseau est sélectionné`awsvpc`, une interface Elastic Network est allouée à la tâche, et vous devez en spécifier une `NetworkConfiguration` lorsque vous créez un service ou exécutez une tâche avec la définition de la tâche. Pour plus d'informations, consultez Mise en réseau des tâches pour les tâches sur les instances Amazon EC2.

Si le mode réseau est `default`, la tâche utilise le réseau virtuel intégré de Docker sous Windows, qui s'exécute à l'intérieur de chaque instance Amazon EC2 hébergeant la tâche. Le réseau virtuel intégré sous Windows utilise le pilote réseau Docker `nat`.

Pour les types de lancement Fargate, lorsque le mode réseau `awsvpc` est activé, une interface Elastic Network est allouée à la tâche, et vous devez en spécifier `NetworkConfiguration` une lorsque vous créez un service ou exécutez une tâche avec la définition de la tâche. Pour plus d'informations, consultez Fargate Task Networking. Le mode réseau `awsvpc` offre les meilleures performances de mise en réseau pour les conteneurs, car ils utilisent la pile de réseau Amazon EC2. Les ports de conteneur exposés sont mappés directement au port attaché de l'interface réseau Elastic. Pour cette raison, vous ne pouvez pas utiliser de mappages de ports hôtes dynamiques.

Les modes réseau `host` et `awsvpc` offrent les meilleures performances de mise en réseau pour les conteneurs, car ils utilisent la pile de réseau Amazon EC2. 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.

Si vous utilisez le type de lancement Fargate, le mode réseau `awsvpc` est requis. Si vous utilisez le type de lancement EC2, le mode réseau dépend du système d'exploitation de l'instance EC2 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 : chaîne

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 des tâches Amazon ECS hébergées sur Fargate sont les suivantes :`LINUX`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE`, `WINDOWS_SERVER_2022_FULL` et `WINDOWS_SERVER_2022_CORE`.

Les valeurs valides des tâches Amazon ECS hébergées sur EC2 sont les suivantes : `LINUX`, `WINDOWS_SERVER_2022_CORE`, `WINDOWS_SERVER_2022_FULL`, `WINDOWS_SERVER_2019_FULL`, et `WINDOWS_SERVER_2019_CORE`, `WINDOWS_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 : chaîne

Obligatoire : Conditionnelle

Par défaut : X86_64

Ce paramètre est requis pour les tâches Amazon ECS hébergées sur Fargate. 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 pour le type de lancement Fargate ou le type de lancement EC2, vous pouvez définir la valeur sur `ARM64`. Pour plus d'informations, consultez the section called “Utilisation des charges de travail ARM 64 bits sur Amazon ECS”.

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 instances Amazon EC2, ces champs sont facultatifs. 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.

> **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.

Le paramètre suivant est autorisé dans une définition de tâche :

`cpu`

Type : chaîne

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. Il peut être exprimé à l'aide d'unités CPU (par exemple,`1024`) ou à l'aide de vCPU (par exemple, `1 vCPU` ou `1 vcpu`) dans une définition de tâche. Lorsque la définition de tâche est enregistrée, la valeur de vCPU (processeurs virtuels) est convertie en un nombre entier indiquant les unités UC.

Pour les tâches qui s'exécutent sur des instances EC2 ou externes, 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 pour les tâches exécutées sur des instances EC2 ou externes se situent entre les vCPU `0.125` et les vCPU `10`.

Pour les tâches exécutées sur Fargate (containers Linux et Windows), ce champ est obligatoire et vous devez utiliser l'une des valeurs suivantes, qui détermine la plage de valeurs prises en charge pour le paramètre `memory`. 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
8192 (8 vCPU)	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
Note Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure		
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 : chaîne

Obligatoire : Conditionnelle

> **Note**
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

Limite stricte (en Mio) de la quantité de mémoire à présenter à la tâche. Il peut être exprimé à l'aide de MiB (par exemple`1024`) ou de Go (par exemple `1GB` ou`1 GB`) dans une définition de tâche. Lorsque la définition de tâche est enregistrée, la valeur en Go est convertie en un nombre entier indiquant la quantité de Mio.

Pour les tâches hébergées sur des instances Amazon EC2, 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 plus d'informations, consultez Gestion de la mémoire des instances de conteneurs.

Si vos tâches sont hébergées sur Fargate (Linux et Windows), ce champ est obligatoire et vous devez utiliser l'une des valeurs suivantes, qui détermine la plage de valeurs prises en charge pour le paramètre `cpu` :

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	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
Note Cette option nécessite la 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 : chaîne

Obligatoire : non

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 : chaîne

Obligatoire : non

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 se mappe à `Image` dans la section Create a container (Création d'un conteneur) de Docker Remote API (L'API Docker à distance) et le paramètre `IMAGE` de 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.
- Les images des registres privés sont prises en charge. Pour plus d'informations, consultez Authentification de registre privé pour les tâches.
- 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`).

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 à `Memory` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--memory` correspond à docker run.

Si vous utilisez le type de lancement Fargate, ce paramètre est facultatif.

Si vous utilisez le type de lancement EC2, vous devez spécifier une valeur de mémoire au niveau de la tâche ou une valeur de mémoire 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 Gestion de la mémoire des instances de conteneurs.

`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 à `MemoryReservation` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--memory-reservation` correspond à 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 Gestion de la mémoire des instances de conteneurs.

Mappages de port

`portMappings`

Type : tableau d'objets

Obligatoire : non

Les mappages de ports permettent aux conteneurs d'accéder aux ports sur l'instance du conteneur hôte pour le trafic entrant ou sortant.

Pour les définitions de tâche qui utilisent le mode réseau `awsvpc`, spécifiez uniquement `containerPort`. Le paramètre `hostPort` peut rester vide ou comporter la même valeur que `containerPort`.

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`) correspond à `PortBindings` dans la section Créer un conteneur de l'API Docker à distance et l'option `--publish` correspond à 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 plus d'informations, consultez 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.

Si vous utilisez des conteneurs dans une tâche avec le type de lancement Fargate, les ports exposés doivent être spécifiés avec `containerPort`.

Pour les conteneurs Windows sur Fargate, vous ne pouvez pas utiliser le port 3 150 pour `containerPort`. C'est parce qu'il est réservé.

Supposons que vous utilisiez des conteneurs dans une tâche ayant le type de lancement EC2 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 plus d'informations, consultez `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 à la fois pour les types de lancement EC2 et AWS Fargate.
- 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 plus d'informations, consultez 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.

Si vous utilisez des conteneurs dans une tâche ayant le type de lancement Fargate, le paramètre `hostPort` peut rester vide ou comporter la même valeur que `containerPort`.

Supposons que vous utilisiez des conteneurs dans une tâche ayant le type de lancement EC2. 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 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 : chaîne

Obligatoire : non, requis pour que Service Connect soit configuré dans un service

Nom utilisé pour le mappage de port. Ce paramètre s'applique uniquement à Service Connect. Ce paramètre correspond au nom que vous utilisez dans la configuration Service Connect d'un service.

Pour plus d'informations, consultez Service Connect.

Dans l'exemple suivant, les deux champs obligatoires pour Service Connect 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 plus d’informations, consultez Authentification de registre privé pour les tâches.

`credentialsParameter`

Type : chaîne

Obligatoire : oui, lorsque des objets `repositoryCredentials` sont utilisés

L'Amazon Resource Name (ARN) du secret contenant les informations d'identification du référentiel privé.

Pour plus d'informations, consultez Authentification de registre privé pour les tâches.

> **Note**
>
> Lorsque vous utilisez l'API Amazon ECS AWS CLI, ou AWS les SDK, 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 étendent les capacités de la commande docker run qui est utilisée pour lancer des conteneurs sur vos instances de conteneur Amazon ECS.

Rubriques

- Surveillance de l'état
- Environnement
- Paramètres réseau
- Stockage et journalisation
- Sécurité

- Limites des ressources
- Étiquettes Docker

Surveillance de l'état

`healthCheck`

Commande de surveillance de l'état du conteneur et paramètres de configuration associés pour le conteneur. Pour plus d'informations, consultez 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 panneau AWS Management Console JSON AWS CLI, ou des API, 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. Pour plus d'informations, consultez `HealthCheck` dans la section Create a container (Créer un conteneur) de Docker Remote API.

`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.

Environnement

`cpu`

Type : entier

Obligatoire : non

Le nombre d'unités `cpu` que l'agent de conteneur Amazon ECS réserve pour le conteneur. Sur Linux, ce paramètre correspond à `CpuShares` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--cpu-shares` correspond à docker run.

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.

> **Note**
>
> Vous pouvez déterminer le nombre d'unités de processeur qui sont disponibles pour chaque type d'instance Amazon EC2. Pour ce faire, multipliez par 1 024 le nombre de vCPU répertoriés pour ce type d'instance sur la page détaillée Instances Amazon EC2.

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. Pour plus d'informations, consultez la section CPU share constraint dans la documentation Docker. La valeur de parts d'UC minimale autorisée par le noyau Linux est de 2. Toutefois, le paramètre UC n'est pas obligatoire et vous pouvez utiliser des valeurs d'UC inférieures à 2 dans vos définitions de conteneur. Pour les valeurs d'UC inférieures à 2 (y compris null), le comportement varie selon la version de l'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..

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 GPU réservés pour tous les conteneurs dans une tâche ne doit pas dépasser le nombre de GPU disponibles sur l'instance de conteneur sur laquelle la tâche est lancée. Pour plus d'informations, consultez Utilisation des GPU sur Amazon ECS.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs Windows ou les conteneurs hébergés sur Fargate.

`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 plus d'informations, consultez the section called "Nom de l'accélérateur Elastic Inference".

> **Note**
>
> À compter du 15 avril 2023, AWS nous n'intégrerons pas de nouveaux clients à Amazon Elastic Inference (EI) et nous aiderons les clients actuels à migrer leurs charges de travail vers des options offrant un meilleur prix et de meilleures performances. Après le 15 avril 2023, les nouveaux clients ne pourront plus lancer d'instances avec les accélérateurs Amazon EI sur Amazon SageMaker, Amazon ECS ou Amazon EC2. Toutefois, les clients qui ont utilisé Amazon EI au moins une fois au cours des 30 derniers jours sont considérés comme des clients actuels et pourront continuer à utiliser le service.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs Windows ou les conteneurs hébergés sur Fargate.

`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 plus d'informations, consultez Architecture de votre application.

```
"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. Ce paramètre correspond à `Entrypoint` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--entrypoint` correspond à docker run. Pour plus d'informations sur le paramètre Docker `ENTRYPOINT`, veuillez consulter https://docs.docker.com/engine/reference/builder/#entrypoint.

```
"entryPoint": ["string", ...]
```

`command`

Type : tableau de chaînes

Obligatoire : non

La commande transmise au conteneur. Ce paramètre correspond à `Cmd` dans la section Create a container (Création d'un conteneur) de Docker Remote API (API distante Docker) et au paramètre `COMMAND` de docker run. Pour plus d'informations sur le paramètre Docker `CMD`, veuillez consulter https://docs.docker.com/engine/reference/builder/#cmd. 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 se mappe à l'option `--env-file` sur docker run.

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. Pour de plus amples informations sur la syntaxe appropriée du fichier de variable d'environnement, consultez Déclarer les variables d'environnement par défaut dans le fichier.

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 plus d'informations, consultez Utiliser les paramètres de définition des tâches pour transmettre des variables d'environnement à un conteneur.

`value`

Type : chaîne

Obligatoire : oui

L'Amazon Resource Name (ARN) de l'objet Amazon S3 contenant le fichier de variable d'environnement.

`type`

Type : chaîne

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 à `Env` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--env` correspond à 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 : chaîne

Obligatoire : oui lorsque `environment` est utilisé

Nom de la variable d'environnement.

`value`

Type : chaîne

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 plus d'informations, consultez Transmission de données sensibles à un conteneur.

`name`

Type : chaîne

Obligatoire : oui

Valeur à définir comme variable d'environnement sur le conteneur.

`valueFrom`

Type : chaîne

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 existe au même Région AWS endroit 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. Ce paramètre correspond à `NetworkDisabled` dans la section Create a container (Crer un conteneur) de Docker Remote API.

> **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. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres, des tirets ou des traits de soulignement. Pour plus d'informations sur la liaison de conteneurs Docker, veuillez consulter https://docs.docker.com/engine/userguide/networking/default_network/dockerlinks/. Ce paramètre correspond à `Links` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--link` correspond à docker run.

> **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 à `Hostname` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--hostname` correspond à 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. Ce paramètre correspond à `Dns` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--dns` correspond à docker run.

> **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 `DnsSearch` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--dns-search` correspond à docker run.

> **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 à `ExtraHosts` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--add-host` correspond à 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 : chaîne

Obligatoire : oui, lorsque des objets `extraHosts` sont utilisés

Nom d'hôte à utiliser dans l'entrée `/etc/hosts`.

`ipAddress`

Type : chaîne

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 `ReadonlyRootfs` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--read-only` correspond à docker run.

> **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 à `Volumes` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--volume` correspond à 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.

`sourceVolume`

Type : chaîne

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : chaîne

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`.

`volumesFrom`

Type : tableau d'objets

Obligatoire : non

Volumes de données à monter à partir d'un autre conteneur. Ce paramètre correspond à `VolumesFrom` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--volumes-from` correspond à docker run.

`sourceContainer`

Type : chaîne

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.

Ce paramètre correspond à `LogConfig` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--log-driver` correspond à `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). Pour plus d'informations sur les options relatives aux différents pilotes du journal pris en charge, veuillez consulter Configurer les pilotes de journalisation dans la documentation Docker (langue française non garantie).

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.
- Pour les tâches utilisant le type de lancement EC2, l'agent de conteneur Amazon ECS en cours d'exécution sur une instance de conteneur doit enregistrer les pilotes de journalisation disponibles sur cette instance avec la variable d'environnement `ECS_AVAILABLE_LOGGING_DRIVERS` avant que les conteneurs placés sur cette instance puissent utiliser des options de configuration de journal. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.
- Pour les tâches utilisant le type de lancement Fargate, tout logiciel supplémentaire nécessaire doit être installé en dehors de la tâche, car vous n'avez pas accès à l'infrastructure sous-jacente sur laquelle vos tâches sont hébergées. 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","logentries","splunk","syslog","awsfirelens",
      "options": {"string": "string"
        ...},
 "secretOptions": [{
  "name": "string",
  "valueFrom": "string"
 }]
}
```

`logDriver`

Type : chaîne

Valeurs valides : `"awslogs","fluentd","gelf","json-file","journald","logentries","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.

Pour les tâches utilisant le type de lancement Fargate, les pilotes de journal pris en charge sont `awslogs`, `splunk` et `awsfirelens`.

Pour les tâches utilisant le type de lancement EC2, les pilotes de journal pris en charge sont `awslogs`, `fluentd`, `gelf`, `json-file`, `journald`, `logentries`, `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, consultezUtilisation du pilote du journal awslogs.

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

La correspondance clé/valeur des options de configuration à envoyer au pilote du journal.

Lorsque vous acheminez les journaux FireLens 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 plus d’informations, consultez the section called “Limite de tampons Fluentd”.

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 plus d'informations, consultez Transmission de données sensibles à un conteneur.

`name`

Type : chaîne

Obligatoire : oui

Valeur à définir comme variable d'environnement sur le conteneur.

`valueFrom`

Type : chaîne

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

FireLens Configuration du conteneur. Cette option est utilisée pour spécifier et configurer un routeur de journal pour les journaux de conteneur. Pour plus d'informations, consultez Utilisation du routage personnalisé des journaux.

```
{
    "firelensConfiguration": {
        "type": "fluentd",
        "options": {
            "KeyName": ""
        }
    }
}
```

`options`

Type : mappage chaîne/chaîne

Obligatoire : non

La correspondance clé/valeur des options à utiliser lors de la configuration du routeur de journal. 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:::mybucket/fluent.conf|filepath"}`. Pour plus d'informations, consultez Spécifier une FireLens configuration dans une définition de tâche.

`type`

Type : chaîne

Obligatoire : oui

Routeur de journal à utiliser. Les valeurs valides sont `fluentd` ou `fluentbit`.

Sécurité

Pour de plus amples informations sur la sécurité du conteneur, consultez Sécurité des tâches et des conteneurs dans le Guide des meilleures pratiques Amazon ECS.

`credentialSpecs`

Type : tableau de chaînes

Obligatoire : non

Une liste d'ARN dans SSM ou Amazon S3 associée à un fichier de spécifications d'informations 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 maximal d'ARN est de 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, veuillez consulter Utilisation de gMSA pour les conteneurs Windows et Utilisation de gMSAs 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 à `Privileged` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--privileged` correspond à 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 à `User` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--user` correspond à 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 de plus amples informations sur les valeurs valides, consultez Configuration de la sécurité d'exécution de Docker. Ce champ n'est pas valide pour les conteneurs dans les tâches qui utilisent le type de lancement Fargate.

Pour les tâches Linux sur EC2, ce paramètre peut être utilisé pour référencer des étiquettes personnalisées pour SELinux et des systèmes de sécurité à plusieurs niveaux AppArmor .

Pour toute tâche sur EC2, ce paramètre peut être utilisé pour référencer un fichier de spécification d'informations d'identification qui configure un conteneur pour l'authentification Active Directory. Pour plus d'informations, consultez Utilisation des GMSA pour les conteneurs Windows sur Amazon EC2 et Utilisation gMSA pour les Linux conteneurs sur Amazon EC2.

Ce paramètre correspond à `SecurityOpt` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--security-opt` correspond à docker.

```
"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 plus d'informations, consultez 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 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.

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 `1024` 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 : chaîne

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`.

`softLimit`

Type : entier

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

La limite flexible du type `ulimit`.

Étiquettes Docker

`dockerLabels`

Type : mappage chaîne/chaîne

Obligatoire : non

Correspondance clé/valeur des étiquettes à ajouter au 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.

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 plus d'informations, consultez Création d'une définition de tâche 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 les caractéristiques par défaut et les autres caractéristiques disponibles, veuillez consulter Privilège d'exécution et fonctionnalités Linux dans la référence Docker run (langue française non garantie). 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 à `CapAdd` dans la section Create a container (Créer un conteneur) de Docker Remote API et l'option `--cap-add` à la commande docker run.

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

`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 à `CapAdd` dans la section Create a container (Créer un conteneur) de Docker Remote API et l'option `--cap-add` à la commande 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 à `CapDrop` dans la section Create a container (Créer un conteneur) de Docker Remote API et l'option `--cap-drop` à la commande docker run.

`devices`

Tous les périphériques hôtes à exposer au conteneur. Ce paramètre correspond à `Devices` dans la section Create a container (Créer un conteneur) de Docker Remote API et l'option `--device` à la commande docker run.

> **Note**
>
> Le paramètre `devices` n'est pas pris en charge lorsque vous utilisez le type de lancement Fargate ou des conteneurs Windows.

Type : tableau d'objets Périphérique

Obligatoire : non

`hostPath`

Chemin du périphérique dans l'instance de conteneur hôte.

Type : chaîne

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 est mappé à l'option `--init` de 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 converti en l'option `--memory-swap` de la commande docker run, où la valeur est la somme de la mémoire du conteneur plus la valeur `maxSwap`.

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.

> **Note**
>
> Si vous utilisez des tâches qui ont recours au type de lancement Fargate, le paramètre `maxSwap` n'est pas pris en charge.

`sharedMemorySize`

Valeur correspondant à la taille (en Mio) du volume `/dev/shm`. Ce paramètre est mappé à l'option `--shm-size` de 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

`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 est mappé à l'option `--memory-swappiness` de docker run.

> **Note**
>
> Si vous utilisez des tâches qui ont recours au type de lancement Fargate, le paramètre `swappiness` n'est pas pris en charge.
> 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 est mappé à l'option `--tmpfs` de 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 : chaîne

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.

Pour les tâches Amazon ECS hébergées sur des instances Amazon EC2, ces instances nécessitent au minimum la 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 plus d'informations, consultez AMI optimisée pour Amazon ECS.

Pour les tâches Amazon ECS hébergées sur Fargate, ce paramètre exige que la tâche ou le service utilise la version de plateforme `1.3.0` ou une version ultérieure (Linux) ou `1.0.0` (Windows).

```
"dependsOn": [
    {
        "containerName": "string",
        "condition": "string"
    }
]
```

`containerName`

Type : chaîne

Obligatoire : oui

Nom de conteneur qui doit répondre à la condition spécifiée.

`condition`

Type : chaîne

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.

Pour les tâches Amazon ECS hébergées sur Fargate, ce paramètre exige que la tâche ou le service utilise la version de plateforme `1.3.0` ou une version 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.

Pour les tâches Amazon ECS hébergées sur Fargate, ce paramètre exige que la tâche ou le service utilise la version de plateforme `1.3.0` ou une version 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.

Pour les tâches utilisant le type de lancement EC2, si le paramètre `stopTimeout` 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 plus d'informations, consultez AMI optimisée pour Amazon ECS.

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 à `Sysctls` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--sysctl` correspond à 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`, y compris Fargate, 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 plus d'informations, consultez 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 de réseau valides : `Sysctls` qui commencent par `"net.*"`

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 à `OpenStdin` dans

la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--interactive` correspond à 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 à `Tty` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--tty` correspond à docker run.

L'argument par défaut est `false`.

Nom de l'accélérateur Elastic Inference

> **Note**
>
> À compter du 15 avril 2023, AWS nous n'intégrerons pas de nouveaux clients à Amazon Elastic Inference (EI) et nous aiderons les clients actuels à migrer leurs charges de travail vers des options offrant un meilleur prix et de meilleures performances. Après le 15 avril 2023, les nouveaux clients ne pourront plus lancer d'instances avec les accélérateurs Amazon EI sur Amazon SageMaker, Amazon ECS ou Amazon EC2. Toutefois, les clients qui ont utilisé Amazon EI au moins une fois au cours des 30 derniers jours sont considérés comme des clients actuels et pourront continuer à utiliser le service.

L'exigence de la ressource d'accélérateur Elastic Inference pour votre définition de tâche. Pour plus d'informations, consultez Qu'est-ce qu'Amazon Elastic Inference ? dans le guide du développeur Amazon Elastic Inference.

Les paramètres suivants sont autorisés dans une définition de tâche :

`deviceName`

Type : chaîne

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 : chaîne

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.

Si vous utilisez le type de lancement Fargate, les contraintes de placement des tâches ne sont pas prises en charge. Par défaut, les tâches Fargate sont réparties entre les zones de disponibilité.

Pour les tâches qui utilisent le type de lancement EC2, vous pouvez utiliser des contraintes afin de placer les tâches en fonction de la zone de disponibilité, du type d'instance ou d'attributs personnalisés. Pour plus d'informations, consultez 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 plus d'informations, consultez Création d'expressions pour définir des instances de conteneur pour les tâches Amazon ECS.

`type`

Type : chaîne

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 lancement EC2, les instances de conteneur doivent au moins disposer de la version 1.26.0 de l'agent de conteneur et au moins de la version 1.26.0-1 du package `ecs-init` pour activer une configuration 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 plus d'informations, consultez AMI optimisée pour Amazon ECS.

Pour les tâches utilisant le type de lancement Fargate, cette fonction exige que la tâche ou le service utilise la version 1.3.0 ou ultérieure de la plateforme.

> **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 : chaîne

Valeur valeurs : `APPMESH`

Obligatoire : non

Type de proxy. La seule valeur prise en charge est `APPMESH`.

`containerName`

Type : chaîne

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 des instances Fargate ou Amazon EC2. Pour plus d’informations, consultez Volumes Amazon EBS.
- 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 qui leur est nécessaire, au moment où elles en ont besoin. Les volumes Amazon EFS sont pris en charge lorsque vous exécutez des tâches sur des instances Fargate ou Amazon EC2. Pour plus d'informations, consultez Volumes Amazon EFS.

- Volumes FSx for Windows File Server : 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 for Windows File Server avec Amazon ECS, vous pouvez allouer vos tâches Windows avec un stockage de fichiers permanent, distribué, partagé et statique. Pour plus d'informations, consultez Volumes FSx for Windows File Server.

  Les conteneurs Windows sur Fargate ne prennent pas en charge cette option.
- Volumes Docker : volume géré par Docker créé sous l'instance Amazon `/var/lib/docker/volumes` EC2 hôte. 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 instances Amazon EC2. 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. Pour plus d'informations, consultez Utilisation des volumes.
- 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 sur des AWS instances Fargate ou Amazon EC2. Pour utiliser des volumes hôte de montage lié, spécifiez un `host` et éventuellement une valeur `sourcePath` dans votre définition de tâche. Pour plus d'informations, consultez Utilisation de montages liés.

Pour plus d'informations, consultez Utilisation de volumes de données dans des tâches.

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 paramètre `host` est utilisé pour lier le cycle de vie du montage lié à l'instance Amazon EC2 hôte, plutôt qu'à la tâche et à l'endroit où elle est stockée. 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 est pris en charge uniquement lors de l'utilisation de tâches hébergées sur des instances Amazon EC2.

`sourcePath`

Type : chaîne

Obligatoire : non

Lorsque le paramètre `host` est utilisé, spécifiez un paramètre `sourcePath` pour déclarer le chemin d'accès sur l'instance Amazon EC2 hôte qui est présentée au conteneur. Si ce paramètre est vide, le démon Docker attribue un chemin hôte pour vous. Si le paramètre `host` contient un emplacement de fichier `sourcePath`, le volume de données persiste à l'emplacement spécifié sur l'instance Amazon EC2 hôte jusqu'à ce que vous le supprimiez manuellement. Si la valeur `sourcePath` n'existe pas sur l'instance Amazon EC2 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 la section Amazon EBS volumes.

`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 instances EC2. 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 : chaîne

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 ou défini sur. `false`

`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. Pour plus d'informations, veuillez consulter Découverte de plug-ins Docker. Ce paramètre correspond à `Driver` dans la section Create a volume (Créer un volume) de Docker Remote API et mappe l'option `--driver` à `docker volume create`.

`driverOpts`

Type : chaîne

Obligatoire : non

Une carte des options spécifiques au pilote Docker à utiliser. Ce paramètre correspond à `DriverOpts` dans la section Create a volume (Créer un volume) de Docker Remote API et mappe l'option `--opt` à `docker volume create`.

`labels`

Type : chaîne

Obligatoire : non

Métadonnées personnalisées à ajouter à votre volume Docker. Ce paramètre correspond à `Labels` dans la section Create a volume (Créer un volume) de Docker Remote API et mappe l'option `--label` à `docker volume create`.

`efsVolumeConfiguration`

Type : VolumeConfiguration objet EFS

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Amazon EFS.

`fileSystemId`

Type : chaîne

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 : chaîne

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 : AuthorizationConfiguration objet EFS

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 : chaîne

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 : SxWindowsFileServerVolumeConfiguration Objet F

Obligatoire : oui

Ce paramètre est spécifié lorsque vous utilisez un système de fichiers Amazon FSx for Windows File Server pour le stockage des tâches.

`fileSystemId`

Type : chaîne

Obligatoire : oui

ID du système de fichiers FSx for Windows File Server à utiliser.

`rootDirectory`

Type : chaîne

Obligatoire : oui

Répertoire du système de fichiers FSx for Windows File Server à monter en tant que répertoire racine à l'intérieur de l'hôte.

`authorizationConfig`

`credentialsParameter`

Type : chaîne

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 : chaîne

Obligatoire : oui

Nom de domaine complet hébergé par un annuaire AWS Directory Service for Microsoft Active Directory(AWS Managed Microsoft AD) ou un annuaire EC2 Active Directory auto-hébergé.

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 plus d'informations, consultez 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 plus d'informations, consultez Création d'une définition de tâche à l'aide de la console.

Rubriques

- Stockage éphémère
- Mode IPC
- Mode PID

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 plus d'informations, consultez the section called "Montages liés".

> **Note**
>
> Ce paramètre est uniquement pris en charge pour les tâches hébergées sur AWS Fargate utilisant la version de plateforme `1.4.0` ou ultérieure (Linux) ou `1.0.0` ou ultérieure (Windows).

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 `host` est spécifié, tous les conteneurs des tâches ayant spécifié le mode IPC `host` 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. Pour plus d'informations, consultez la section IPC settings (Paramètres IPC) sur la page Docker run reference.

Si le mode IPC `host` est utilisé, il existe un risque accru d'exposition à un espace de noms IPC indésirable. Pour plus d'informations, consultez Sécurité Docker.

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 plus d'informations, consultez Contrôles système.

- 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 : chaîne

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 Fargate pour 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 `host` est spécifié, tous les conteneurs des tâches ayant spécifié le mode PID `host` sur la même instance de conteneur partagent le même espace de noms de processus avec l'instance Amazon EC2 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. Pour plus d'informations, consultez la section PID settings (Paramètres PID) sur la page Docker run reference.

Si le mode PID `host` est utilisé, il existe un risque accru d'exposition à un espace de noms de processus indésirable. Pour plus d'informations, consultez Sécurité Docker.

> **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.

Modèle de définition de tâche

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-json`option. Pour plus d'informations, consultez Paramètres de définition de tâche.

Modèle de type de lancement Amazon EC2

```
{
    "family": "",
    "taskRoleArn": "",
    "executionRoleArn": "",
    "networkMode": "none",
    "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": "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"
                }
            }
        }
    ],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": ""
        }
    ],
    "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
```

Paramètres de définition de service

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 : chaîne

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 plus d'informations, consultez 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 : chaîne

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 plus d’informations, consultez Versions de plateforme Linux Fargate.

> **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 : chaîne

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 : chaîne

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 plus d'informations, consultez Concepts du 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 contraintes de placement des tâches afin de personnaliser la façon dont les tâches sont placées. Pour plus d'informations, consultez Replica.

- `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 plus d'informations, consultez Démon.

  > **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.

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 service `REPLICA` et dispose d'un `desiredCount` de quatre tâches, ainsi que d'une valeur `maximumPercent` de 200 %, le planificateur pourra 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 `maximumPercent` par défaut pour un service à l'aide du planificateur de service `REPLICA` est de 200 %.

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 les types de déploiement bleu/vert (`CODE_DEPLOY`) ou `EXTERNAL` et des tâches employant le type de lancement EC2, la valeur de pourcentage maximal est définie sur la valeur par défaut et est utilisée pour définir la limite maximale du nombre de tâches dans le service qui restent à l'état `RUNNING` pendant que les instances de conteneur sont à l'état `DRAINING`. Si les tâches dans le service utilisent le type de lancement Fargate, la valeur de pourcentage maximal n'est pas utilisée, même si elle est renvoyée pour décrire 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 a un `desiredCount` de quatre tâches et un `minimumHealthyPercent` de 50 %, le planificateur de service

peut arrêter deux tâches existantes pour libérer de la capacité de cluster avant de lancer deux nouvelles tâches.

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 plus d’informations, consultez 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 de 0 % pour le AWS CLI, AWS les SDK et les API et de 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 les types de déploiement bleu/vert (`CODE_DEPLOY`) ou `EXTERNAL` et des tâches employant le type de lancement EC2, la valeur de pourcentage minimum d'instances

saines est définie sur la valeur par défaut et est utilisée pour définir la limite inférieure du nombre de tâches dans le service qui restent à l'état `RUNNING` pendant que les instances de conteneur sont à l'état `DRAINING`. 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 plus d'informations, consultez Types de déploiement Amazon ECS.

`type`

Type : chaîne

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 bleu/vert (`CODE_DEPLOY`) utilise le modèle de déploiement bleu/vert alimenté par CodeDeploy, qui vous permet de vérifier le 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 plus d'informations, consultez 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 : chaîne

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 plus d'informations, consultez Balisage des ressources Amazon ECS.

`key`

Type : chaîne

Contraintes de longueur : Longueur minimum de 1. Longueur maximum 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 : chaîne

Contraintes de longueur : Longueur minimum de 0. Longueur maximum 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 plus d'informations, consultez Identification de vos ressources pour facturation.

`propagateTags`

Type : chaîne

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 lancement Amazon EC2, consultez. Mise en réseau des tâches pour les tâches sur les instances Amazon EC2 Pour plus d'informations sur la mise en réseau pour le type de lancement Fargate, consultez la section Mise en réseau des tâches 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 : chaîne

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 service Amazon ECS service doit ignorer les surveillances de l'état des cibles Elastic Load Balancing non saines, les surveillances de l'état du conteneur et les surveillances de l'état de Route 53 après le passage d'une tâche à l'état `RUNNING`. Cela est valable uniquement si votre service est configuré pour utiliser un équilibreur de charge. Si votre service dispose d'un équilibreur de charge défini et que vous ne spécifiez pas de valeur de période de grâce de surveillance de l'état, c'est la valeur par défaut de `0` qui est utilisée.

Si des tâches de votre service prennent du temps pour démarrer et répondre aux surveillances de l'état, vous pouvez spécifier une période de grâce pouvant atteindre 2 147 483 647 secondes,

pendant laquelle le planificateur de service ECS ignore le statut de la surveillance de l'état. Cette période de grâce peut empêcher le planificateur de service ECS de marquer les tâches comme étant non saines et de les arrêter avant qu'elles aient le temps de s'exécuter.

Si vous n'utilisez pas un Elastic Load Balancing, nous vous recommandons d'utiliser `startPeriod` dans les paramètres du contrôle de surveillance de l'état de la définition de la tâche. Pour plus d'informations, consultez Health check (Surveillance de l'état).

`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 façon 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 qui utilisent le type de lancement EC2, l'instance de conteneur sur laquelle elles sont lancées doit autoriser le trafic entrant sur le `hostPort` du mappage de port.

`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 plus d'informations, consultez 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 plus d'informations, consultez Service Connect.

`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 : chaîne

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 plus d'informations, consultez Découverte de service.

`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.

Configurations de volume

`volumeConfigurations`

Type : objet

Obligatoire : non

Configuration qui sera utilisée pour créer des volumes pour les tâches gérées par le service. Un volume est créé pour chaque tâche du 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 de données Amazon EBS à des tâches gérées par un service. Pour plus d'informations, consultez la section Amazon EBS volumes.

`name`

Type : chaîne

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 des volumes Amazon EBS associés à des tâches gérées par un service lors de la création ou de la mise à jour d'un service.

`encrypted`

Type : booléen

Obligatoire : non

Valeurs valides : `true`|`false`

Spécifie si le volume Amazon EBS associé aux tâches gérées par un service sera chiffré. Si vous avez activé le chiffrement Amazon EBS par défaut pour votre compte, ce paramètre sera remplacé et le volume sera chiffré. Pour plus d'informations sur le chiffrement EBS par défaut, consultez la section Chiffrement par défaut dans le guide de l'utilisateur Amazon EC2.

`kmsKeyId`

Type : chaîne

Obligatoire : non

Identifiant de la clé AWS Key Management Service (AWS KMS) à utiliser pour le chiffrement Amazon EBS. Si ce paramètre n'est pas spécifié, c'est votre AWS KMS key pour Amazon EBS qui est utilisé. Si `KmsKeyId` est spécifié, l'état chiffré doit être `true`.

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 du volume EBS. Pour plus d'informations sur les types de volumes, consultez les types de volumes Amazon EBS dans le guide de l'utilisateur Amazon EC2. Le type de volume par défaut est `gp3`.

> **Note**
>
> Le type de `standard` volume n'est pas pris en charge pour les volumes Amazon EBS configurés pour être joints à des tâches Fargate.

`sizeInGiB`

Type : entier

Obligatoire : non

Plage valide : entiers compris entre 1 et 16 384

Taille du volume EBS en gibioctets (GiB). Si vous ne fournissez pas d'ID de capture d'écran pour configurer un volume à 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

ID de l'instantané d'un volume EBS existant utilisé pour créer un nouveau volume attaché à la tâche ECS.

iops

Type : entier

Obligatoire : non

Le nombre d'opérations d'E/S 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 crédits d'E/S 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 à des tâches gérées par un service.

> **Important**
>
> Ce paramètre n'est pris en charge que pour les `gp3` volumes.

`roleArn`

Type : chaîne

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 plus d'informations, consultez Rôle IAM dans l'infrastructure Amazon ECS.

`tagSpecifications`

Type : objet

Obligatoire : non

Spécification des balises à appliquer aux volumes Amazon EBS gérés par le service.

`resourceType`

Type : chaîne

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. `AmazonECSCreated`et `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 plus d'informations, consultez Balisage des ressources Amazon ECS.

`key`

Type : chaîne

Contraintes de longueur : Longueur minimum de 1. Longueur maximum 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 : chaîne

Contraintes de longueur : Longueur minimum de 0. Longueur maximum 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 : chaîne

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`

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. La valeur par défaut pour les volumes attachés à des tâches Linux estXFS.

Modèle de définition de service

Voici une illustration de la représentation JSON d'une définition de service Amazon ECS service.

Type de lancement d'Amazon EC2

```
{
    "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,
    "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": "",
                "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
        }
    },
    "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,
    "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": "",
                "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
```

Quotas de service Amazon ECS service

Le tableau suivant fournit les quotas de service par défaut, également appelés limites, pour Amazon ECS pour un Compte AWS. Pour plus d'informations sur les quotas de service pour les autres Services AWS que vous pouvez utiliser avec Amazon ECS, tels qu'Elastic Load Balancing et Auto Scaling, veuillez consulter Service Quotas AWS 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.

Service Quotas Amazon ECS

Voici les Service Quotas Amazon ECS service.

AWS Les nouveaux comptes peuvent avoir des quotas initiaux inférieurs qui peuvent augmenter au fil du temps. Amazon ECS surveille en permanence l'utilisation du compte dans chaque région, puis augmente automatiquement les quotas en fonction de votre utilisation. Vous pouvez également demander une augmentation du quota pour les valeurs affichées comme ajustables, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

Nom	Par défaut	Ajusta	Description
Fournisseurs de capacités par cluster	Chaque Région prise en charge : 20	Non	Nombre maximal de fournisseurs de capacité pouvant être associés à un cluster.
Classic Load Balancers par service	Chaque région prise en charge : 1	Non	Nombre maximal de Classic Load Balancers par service.
Clusters par compte	Chaque région prise en charge : 10 000	Oui	Nombre de clusters par compte
Instances de conteneur par cluster	Chaque région prise en charge : 5 000	Non	Nombre d'instances de conteneur par cluster

Nom	Par défaut	Ajusta	Description
Instances de conteneur par tâche de démarrage	Chaque région prise en charge : 10	Non	Le nombre maximum d'instances de conteneur spécifiées dans une action d' StartTask API.
Conteneurs par définition de tâche	Chaque région prise en charge : 10	Non	Nombre maximal de définitions de conteneur s dans une définition de tâche.
Sessions ECS Exec	Chaque Région prise en charge : 1 000	Non	Nombre maximal de sessions ECS Exec par conteneur.
Taux de tâches lancées par un service sur AWS Fargate	Chaque région prise en charge : 500	Non	Nombre maximal de tâches pouvant être provisionnées par service par minute sur Fargate par le planificateur d'Amazon ECS service.
Taux de tâches lancées par un service sur une instance Amazon EC2 ou une instance externe	Chaque région prise en charge : 500	Non	Nombre maximal de tâches pouvant être provisionnées par service par minute sur une instance Amazon EC2 ou externe par le planifica teur d'Amazon ECS service.

Nom	Par défaut	Ajusta	Description
Révisions par famille de définition de tâche	Chaque région prise en charge : 1 000 000	Non	Nombre maximal de révisions par famille de définition de tâche. Le désenregistrement ou la suppression d'une révision de définition de tâche ne l'exclut pas de cette limite.
Groupes de sécurité par awsvpcCon figuration	Chaque région prise en charge : 5	Non	Nombre maximum de groupes de sécurité spécifiés dans un awsvpcConfiguration.
Services par cluster	Chaque région prise en charge : 5 000	Oui	Nombre maximum de services par cluster
Services par espace de noms	Chaque Région prise en charge : 100	Oui	Nombre maximum de services pouvant être exécutés dans un espace de noms.
Sous-réseaux par awsvpcConfiguration	Chaque région prise en charge : 16	Non	Nombre maximal de sous-réseaux spécifiés dans un awsvpcCon figuration.
Étiquettes par ressource	Chaque région prise en charge : 50	Non	Nombre maximal d'étiquettes par ressource . Cela s'applique aux définitions de tâche, aux clusters, aux tâches et aux services.

Nom	Par défaut	Ajusta	Description
Groupes cibles par service	Chaque région prise en charge : 5	Non	Nombre maximal de groupes cibles par service, si vous utilisez un Application Load Balancer ou un Network Load Balancer.
Taille de la définition de tâche	Toutes les Régions prises en charge : 64 kilo-octets	Non	Taille maximale, en Kio, d'une définition de tâche.
Tâches en état PROVISIONING par cluster	Chaque région prise en charge : 500	Non	Nombre maximal de tâches en attente dans l'état PROVISIONING par cluster. Ce quota s'applique uniquemen t aux tâches lancées à l'aide d'un fournisseur de capacité de groupe EC2 Auto Scaling.
Tâches lancées par run-task	Chaque région prise en charge : 10	Non	Nombre maximal de tâches pouvant être lancées par action d' RunTask API.
Tâches par service	Chaque région prise en charge : 5 000	Non	Nombre maximal de tâches par service (nombre souhaité).

Note

Les valeurs par défaut sont les quotas initiaux définis par AWS, qui sont distincts de la valeur de quota réellement appliquée et du quota de service maximal possible. Pour plus

d'informations, veuillez consulter la rubrique Terminologie des Service Quotas dans le Guide de l'utilisateur Service Quotas.

> **Note**
>
> Les services configurés pour utiliser la découverte de services Amazon ECS service sont limités à 1 000 tâches par service. Cela est dû au quota de service AWS Cloud Map pour le nombre d'instances par service. Pour plus d'informations, consultez la section Quotas du service AWS Cloud Map dans le Référence générale d'Amazon Web Services.

> **Note**
>
> Dans la pratique, les taux de lancement des tâches 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 auprès d'un équilibreur de charge. Vous verrez peut-être des variations dans les taux de lancement des tâches par rapport aux quotas représentés ici. Ces variations sont dues aux fonctionnalités que vous avez activées pour vos services Amazon ECS. Pour plus d'informations, consultez Accélération des déploiements Amazon ECS dans le Guide des bonnes pratiques Amazon ECS.

> **Note**
>
> Les services configurés pour utiliser Amazon ECS Service Connect sont limités à 1 000 tâches par service. Cela est dû au quota AWS Cloud Map de service pour le nombre d'instances par service. Pour plus d'informations, consultez la section Quotas du service AWS Cloud Map dans le Référence générale d'Amazon Web Services.

AWS Fargate quotas de service

Les informations suivantes concernent Amazon ECS sur les quotas de AWS Fargate service et sont répertoriées sous le AWS Fargateservice dans la console Service Quotas.

AWS Les nouveaux comptes peuvent avoir des quotas initiaux inférieurs qui peuvent augmenter au fil du temps. Fargate surveille en permanence l'utilisation du compte dans chaque région, puis augmente automatiquement les quotas en fonction de votre utilisation. Vous pouvez également demander une augmentation du quota pour les valeurs affichées comme ajustables, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

Nom	Par défaut	Ajusta	Description
Nombre de ressources vCPU Fargate à la demande	Chaque région prise en charge : 6	Oui	Nombre de vCPU Fargate qui s'exécutent simultanément en tant que Fargate On-Demand dans ce compte pour la région actuelle.
Nombre de ressources Fargate Spot vCPU	Chaque région prise en charge : 6	Oui	Nombre de vCPU Fargate qui s'exécutent simultanément en tant que Fargate Spot dans ce compte pour la région actuelle.

Note

Les valeurs par défaut sont les quotas initiaux définis par AWS, qui sont distincts de la valeur de quota réellement appliquée et du quota de service maximal possible. Pour plus d'informations, veuillez consulter la rubrique Terminologie des Service Quotas dans le Guide de l'utilisateur Service Quotas.

Note

Fargate applique également les tâches Amazon ECS et les limites de taux de lancement des pods Amazon EKS. Pour plus d'informations, consultez Limitations Fargate.

Gestion de votre Amazon ECS et de vos quotas AWS Fargate de service dans le 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 de plus amples informations, veuillez consulter What Is 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 à l'adresse 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.

Bonnes pratiques relatives aux quotas de service Amazon ECS et aux 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 instances Amazon EC2, le nombre maximum d'instances Amazon EC2 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 instances Amazon EC2 pour 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 instances Amazon EC2 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, consultezRépartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge.

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 qui correspond au 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.

Équilibreur de charge du réseau

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 s'accompagne de limitations de dimensionnement supplémentaires concernant `Targets per Availability Zone Per Network Load Balancer` le 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 `RegisterTarget`, `DeregisterTarget`, 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 AWS 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. Bonnes pratiques pour 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 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 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 instances Amazon EC2, lors de l'exécution de tâches utilisant `awsvpc` le mode réseau, tenez compte `Maximum network interfaces` du quota de service, c'est-à-dire du nombre maximum d'instances réseau pour chaque instance Amazon EC2. 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 l'utilisateur Amazon EC2.

Bien que vous ne puissiez pas modifier le nombre d'interfaces réseau pouvant être associées à une instance Amazon EC2, 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 plus d'informations, consultez Jonction d'interface réseau Elastic.

Limitation des API d'interface réseau élastique

Lorsque vous exécutez des tâches qui utilisent le mode `awsvpc` réseau, Amazon ECS s'appuie sur les API Amazon EC2 suivantes. Chacune de ces API possède des limites d'API différentes. Pour plus d'informations, consultez la section Régulation des demandes pour l'API Amazon EC2 dans le manuel Amazon EC2 API Reference.

- CreateNetworkInterface

- AttachNetworkInterface
- DetachNetworkInterface
- DeleteNetworkInterface
- DescribeNetworkInterfaces
- DescribeVpcs
- DescribeSubnets
- DescribeSecurityGroups
- DescribeInstances

Si les appels d'API Amazon EC2 sont limités pendant les flux de travail de provisionnement d'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 d'API Amazon EC2, vous pouvez nous contacter AWS Support pour obtenir des conseils sur la manière d'augmenter vos limites de limitation d'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 utilisent AWS Cloud Map des API 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 de tâches que vous pouvez exécuter à un maximum de 1 000 tâches de service. Vous ne pouvez pas modifier le AWS Cloud Map quota. Pour de plus amples informations, veuillez consulter les Service Quotas AWS Cloud Map.

AWS Cloud Map Limitation de l'API

Amazon ECS appelle les `DeregisterInstance` AWS Cloud Map API `ListInstances` `GetInstancesHealthStatusRegisterInstance`,, et 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 ces API. Si vous êtes confronté à un ralentissement dû à ces derniers, vous pouvez nous contacter AWS 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. Bonnes pratiques pour gérer les problèmes de régulation d'Amazon ECS

Référence d'API Amazon ECS

Outre l'AWS Management Console et l'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 sur l'AWS CLI, veuillez consulter Référence de l'AWS Command Line Interface pour Amazon ECS (langue française non garantie).

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
GMSA pour les conteneurs 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'informations, voir 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 CloudWatch 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 politique AmazonECS. InfrastructureRolePolicyForServiceConnectTransportLayerSecurity	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` et `perRequestTimeout` .	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 plus d'informations, consultez Systèmes d'exploitation et architectures système pris en charge.	16 janvier 2024
Ajouter une `AmazonECSInfrastructureRolePolicyForVolumes` politique IAM	Le `AmazonECSInfrastructureRolePolicyForVolumes` 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 ServiceRolePolicy gérée par AmazonECS a été mise à jour avec de nouvelles `events` autorisations et autorisations supplémentaires`autoscaling` . `autoscaling-plans`	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) dans le guide de l'utilisateur Amazon ECS pour Fargate. AWS	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, veuillez consulter Utilisation de gMSAs pour les conteneurs Linux et Utilisation de gMSA 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

Modification	Description	Date
Disponibilité régionale mise à jour pour la prise en charge d'ARM64 par Fargate	La disponibilité régionale pour la prise en charge d'ARM64 par Fargate a été mise à jour. Pour plus d'informations, consultez Considérations.	19 juin 2023
Amélioration de la documentation relative à l'autoscaling de cluster	La documentation relative à la mise à l'échelle 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 `ecsCreate Cluster` . 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
Prise en charge de gMSA pour des conteneurs Linux sur EC2	Vous pouvez utiliser gMSA pour vous authentifier auprès d'Active Directory pour des conteneurs Linux sur EC2. Pour plus d'informations, veuillez consulter Utilisation de gMSAs 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

Modification	Description	Date
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 plus d'informations, consultez Rapports d'utilisation et de coût au niveau des tâches.	12 avril 2023
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 plus d'informations, consultez AMI optimisée pour Amazon ECS.	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 plus d'informations, consultez 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

Modification	Description	Date
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 plus d'informations, consultez Amazon ECS_ FullAccess.	4 janvier 2023
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 plus d'informations, consultez the section called "Méthodes de détection des échecs".	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 plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez 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 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 plus d'informations, consultez 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 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 service" et the section called "Exécuter une application en tant que tâche Amazon ECS".	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 plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez Utilisation 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 the section called "AWS Fargate quotas de service".	8 septembre 2022

Modification	Description	Date
Prise en charge des groupes d'instances pré-initialisées pour Amazon EC2 Auto Scaling.	Vous pouvez désormais utiliser les groupes d'instances pré-initialisées Amazon EC2 Auto Scaling pour monter vos applications en puissance plus rapidement et réduire les coûts. Pour plus d'informations, consultez Utilisation d'un groupe d'instances pré-initialisées pour votre groupe Auto Scaling.	23 mars 2022
Prise en charge des instances Windows dans ECS Anywhere.	ECS Anywhere prend désormais en charge les instances Windows. Pour plus d'informations, consultez Instances externes (Amazon ECS Anywhere).	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 plus d'informations, consultez 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 pour le type de lancement Fargate et External à l'aide de la console, Suppression d'un cluster avec la console, Mise à jour d'une définition de tâche à l'aide de la console et Annulation de l'enregistrement d'une révision de définition de tâche 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 plus d'informations, consultez Création d'une définition de tâche à l'aide de la console.	23 novembre 2021

Modification	Description	Date
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'exploitation Linux. Pour plus d'informations, consultez the section called "Utilisation des charges de travail ARM 64 bits sur Amazon ECS".	23 novembre 2021
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 plus d'informations, consultez Utilisation du routage personnalisé des journaux.	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 plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez Versions de plateformes 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 Utilisation des GPU sur Amazon ECS et Enregistrement d'une instance externe dans un cluster.	8 octobre 2021

Modification	Description	Date
Prise en charge du mode réseau `awsvpc` sous Windows	Amazon ECS prend en charge `awsvpc` sur Windows. Pour plus d'informations, consultez Mode réseau `awsvpc`.	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 plus d'informations, consultez AMI Bottlerocket optimisées pour Amazon ECS.	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 plus d'informations, consultez 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 plus d'informations, consultez Mise en route avec 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 plus d'informations, consultez Instances externes (Amazon ECS Anywhere).	25 mai 2021
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 plus d'informations, consultez AMI optimisée pour Amazon ECS.	19 avril 2021

Modification	Description	Date
Amazon ECS Exec	Amazon ECS a publié un nouvel outil de débogage appelé ECS Exec. Pour plus d'informations, consultez 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 stratégies de point de terminaison Amazon VPC Pour plus d'informations, consultez 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 service à l'aide de la console et Exécuter 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 les tâches Amazon FSx for Windows File Server pour Windows.	Amazon ECS a ajouté la prise en charge de la spécification de volumes Amazon FSx for Windows File Server dans les définitions de tâches Windows. Pour plus d'informations, consultez Volumes FSx for Windows File Server.	11 novembre 2020
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 réseau `awsvpc`, qui prend en charge les adresses IPv6. Pour plus d'informations, consultez Utilisation d'un VPC en mode double pile.	5 novembre 2020

Modification	Description	Date
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 plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez AMI Bottlerocket optimisées pour Amazon ECS.	31 août 2020
Point de terminaison des métadonnées de tâches version 4 mis à jour pour les statistiques de fréquence réseau	Le point de terminaison de métadonnées de tâche version 4 a été mis à jour pour fournir des statistiques de débit réseau pour les tâches Amazon ECS qui utilisent les modes réseau `awsvpc` ou `bridge` hébergés sur des instances Amazon EC2 exécutant au moins la version `1.43.0` de l'agent de conteneur. Pour plus d'informations, consultez Point de terminaison des métadonnées des tâches Amazon ECS, version 4.	10 août 2020

Modification	Description	Date
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 plus d'informations, consultez Métriques d'utilisation.	3 août 2020
AWS Copilote version 0.1.0	La nouvelle AWS CLI Copilot a été lancée, fournissant 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 plus d'informations, consultez Utilisation de l'interface de ligne de commande AWS Copilot.	9 juillet 2020
AWS Fargate calendrier de dépréciation des versions de plate-forme	Le calendrier de l'obsolescence des versions de la plateforme Fargate a été ajouté. Pour plus d'informations, consultez 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
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 plus d'informations, consultez AMI optimisée pour Amazon ECS.	24 juin 2020
Ajout de la prise en charge de la suppression 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

Modification	Description	Date
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 plus d'informations, consultez Stockage éphémère de la tâche Fargate.	28 mai 2020
Prise en charge du fichier de variable d'environnement	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 plus d'informations, consultez Utiliser les paramètres de définition des tâches pour transmettre des variables d'environnement à un conteneur.	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
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 Quotas de service Amazon ECS service.	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 plus d'informations, consultez Volumes Amazon EFS. • Le stockage des tâches éphémères est désormais de 20 Go. Pour plus d'informations, consultez Stockage éphémère de la tâche Fargate. • 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 de plus amples informations, veuillez consulter Mise en réseau des tâches Fargate dans le Guide de l'utilisateur Amazon Elastic Container Service pour AWS Fargate. • Les ENI des tâches incluent désormais la prise en charge des trames 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	8 avril 2020

Modification	Description	Date
	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 plus d'informations, consultez Surveillez les conteneurs Amazon ECS à l'aide de Container Insights. • Ajout de la prise en charge du point de terminaison 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 plus d'informations, consultez 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 plus d'informations, consultez 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 nement d'exécution du conteneur mentionneront de	

Modification	Description	Date
	moins en moins Docker et de en plus des erreurs générales. Pour plus d'informations, consultez Versions de plateforme Linux Fargate.	
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 plus d'informations, consultez Volumes Amazon EFS.	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 plus d'informations, consultez 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 plus d'informa tions, consultez Transmission de données sensibles à un conteneur.	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 plus d'informations, consultez Déploiement bleu/vert avec CodeDeploy.	6 février 2020
Ajout du paramètre de définition de efsVolume Configuration 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 plus d'informations, consultez Volumes Amazon EFS.	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 plus d'informations, consultez 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 plus d'informations, consultez AWS Fargate fournisseurs de capacité.	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 plus d'informations, consultez Auto Scaling de cluster Amazon ECS.	3 décembre 2019
Fournisseurs de capacité de cluster	Les fournisseurs de capacité de cluster Amazon ECS déterminent l'infrastructure à utiliser pour vos tâches. Pour plus d'informations, consultez Fournisseurs de capacité Amazon ECS.	3 décembre 2019

Modification	Description	Date
Création d'un cluster sur un AWS Outposts	Amazon ECS prend désormais en charge la création de clusters sur un AWS Outposts. Pour plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez Utilisation des GPU sur Amazon ECS.	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 plus d'informations, consultez Utilisation du routage personnalisé des journaux.	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. Pour plus d'informations, consultez Les conteneurs Deep Learning Containers avec Elastic Inference sur Amazon ECS.	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 plus d'informations, consultez Utilisation du routage personnalisé des journaux.	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 plus d'informations, consultez Surveillez les conteneurs Amazon ECS à l'aide de Container Insights.	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 plus d'informations, consultez Gestion de l'espace d'échange de conteneur.	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	Prise en charge de types d'instance Amazon EC2 supplémentaires pour la fonction de jonction d'ENI. Pour plus d'informations, consultez Types d'instances Amazon EC2 pris en charge.	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 plus d'informations, consultez Enregistrement de plusieurs groupes cibles auprès d'un service.	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 plus d'informations, consultez Spécification de données sensibles à l'aide de secrets de Secrets Manager.	20 juillet 2019
CloudWatch Informations sur les conteneurs	Amazon ECS a ajouté la prise en charge de CloudWatch Container Insights. Pour plus d'informations, consultez Surveillez les conteneurs Amazon ECS à l'aide de Container Insights.	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 autorisations au niveau des ressources pour les services et ensembles de tâches Amazon ECS. Pour plus d'informations, consultez Fonctionnement d'Amazon Elastic Container Service avec IAM.	27 juin 2019
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

Modification	Description	Date
Jonction Elastic Network Interface	Amazon ECS introduit la prise en charge du lancement des instances de conteneur grâce à des types d'instances Amazon EC2 pris en charge qui ont augmenté la densité d'interface réseau Elastic (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 plus d'informations, consultez Jonction d'interface réseau Elastic.	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 plus d'informations, consultez 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 plus d'informations, consultez Transmission de données sensibles à un conteneur.	1er mai 2019
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 référençant dans la définition de votre conteneur. Pour plus d'informations, consultez Transmission de données sensibles à un conteneur.	2 avril 2019

Modification	Description	Date
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'utiliser n'importe quel contrôleur de déploiement tiers pour un contrôle complet du processus de déploiement pour un service Amazon ECS service. Pour plus d'informations, consultez Utiliser un contrôleur tiers pour les déploiements externes d'Amazon ECS.	27 mars 2019
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 Containers 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 plus d'informations, consultez 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 supplémentaires de définition de tâche qui vous permettent de définir des dépendances pour le démarrage et l'arrêt de conteneurs, ainsi qu'une valeur de temporisation d'arrêt et de lancement pour chaque conteneur. Pour plus d'informations, consultez Dépendances du conteneur.	7 mars 2019

Modification	Description	Date
Amazon ECS présente l'API `PutAccountSettingDefault`	Amazon ECS présente l'API `PutAccountSettingDefault` qui permet à un utilisateur de définir le statut d'acceptation du format ARN/ID 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 plus d'informations, consultez Amazon Resource Names (ARN) et ID.	8 février 2019
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 GPU requis et l'agent ECS épinglera les GPU physiques au conteneur. Pour plus d'informations, consultez Utilisation des GPU sur Amazon ECS.	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 plus d'informations, consultez Transmission de données sensibles à un conteneur.	21 janvier 2019

Modification	Description	Date
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
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 plus d’informations, consultez Transmission de données sensibles à un conteneur. • 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 Maintenance de tâches dans le Guide de l'utilisateur Amazon Elastic Container Service pour AWS Fargate. Pour plus d’informations, consultez Versions de plateforme Linux Fargate.	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 plus d'informations, consultez 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 plus d'informations, consultez Régions prises en charge pour Amazon ECS sur AWS Fargate.	7 décembre 2018
Déploiements bleu/vert Amazon ECS	Amazon ECS a ajouté la prise en charge des déploiements bleu/vert à l'aide de. CodeDeploy Ce type de déploiement vous permet de vérifier un nouveau déploiement d'un service avant de lui envoyer un trafic de production. Pour plus d'informations, consultez Déploiement bleu/vert avec CodeDeploy.	27 novembre 2018
Publication de l'AMI Amazon Linux 2 (arm64) optimisée pour Amazon ECS	Amazon ECS a publié une AMI Amazon Linux 2 optimisée pour Amazon ECS pour l'architecture arm64. Pour plus d'informations, consultez AMI optimisée pour Amazon ECS.	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 plus d'informations, consultez Transmission de données sensibles à un conteneur.	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 plus d'informations, consultez 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 plus d'informations, consultez Amazon ECS sur AWS Fargate.	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 plus d’informations, consultez 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 plus d’informations, consultez Amazon ECS sur AWS Fargate.	26 octobre 2018
Publication d'une AMI Amazon Linux 2 optimisée pour Amazon ECS	Amazon ECS propose deux versions des AMI Linux optimisées pour le service. La version la plus récente et recommandée est basée sur x;. Amazon ECS vend également des AMI basées 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 plus d’informations, consultez AMI optimisée pour Amazon ECS.	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 plus d'informations, consultez Récupérez les 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 plus d'informations, consultez Découverte de service.	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'authentification 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 plus d'informations, consultez Authentification de registre privé pour les tâches.	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 plus d'informations, consultez Découverte de service.	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'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 plus d'informations, consultez Authentification de registre privé pour les tâches.	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 plus d'informations, consultez Utilisation de volumes de données dans des tâches.	9 août 2018
AWS Expansion de la région de Fargate	AWS Fargate with Amazon ECS s'est étendu aux régions Europe (Francfort), Asie-Pacifique (Singapour) et Asie-Pacifique (Sydney). Pour plus d'informations, consultez Amazon ECS sur AWS Fargate.	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 plus d'informations, consultez Replica. • `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 plus d'informations, consultez Démon. **Note** Les tâches Fargate ne prennent pas en charge la stratégie de planification `DAEMON`. Pour plus d'informations, consultez Concepts du planificateur de service.	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 d'une procédure d'installation manuelle de l'agent de conteneur à partir d'une URL S3 sur une instance EC2 autre qu'Amazon Linux. Cette procédure inclut une méthode de signature PGP permettant de vérifier le fichier d'installation de l'agent de conteneur Amazon ECS. Pour plus d'informations, consultez Installation de l'agent de conteneur Amazon ECS. • Ajout d'une procédure pour installer manuellement l'agent de conteneur à partir d'une URL S3 sur une instance EC2 Windows. Cette procédure inclut une méthode de signature PGP pour vérifier le fichier d'installation de l'agent de conteneur Amazon ECS. • 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 plus d'informations, consultez 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 plus d'informations, consultez Découverte de service.	le 22 mai 2018

Modification	Description	Date
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 plus d'informations, consultez Extraction des métadonnées d'AMI optimisée pour Amazon ECS.	le 9 mai 2018
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 plus d'informations, consultez Amazon ECS sur AWS Fargate.	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 plus d'informations, consultez Extraction des métadonnées d'AMI optimisée 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 Récupérez les métadonnées Amazon ECS. • Ajout de la prise en charge de Surveillance de l'état. • Ajout de la prise en charge de Découverte de service Pour plus d'informations, consultez Versions de plateforme Linux Fargate.	26 mars 2018

Modification	Description	Date
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 plus d'informations, consultez Découverte de service.	22 mars 2018
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 plus d'informations, consultez Surveillance de l'état.	8 mars 2018
AWS Fargate	Ajout d'une présentation d'Amazon ECS avec AWS Fargate. Pour plus d'informations, consultez Amazon ECS sur AWS Fargate.	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 plus d'informations, consultez Récupérez les 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 plus d'informations, consultez Faites évoluer votre service Amazon ECS à l'aide d'une valeur métrique cible. 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 plus d'informations, consultez AMI optimisée pour Amazon ECS.	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 documenté qui se déclenche après deux échecs de lancement consécutifs d'une tâche du service. Pour plus d'informations sur ce nouveau comportement, consultez Concepts du service supplémentaire.	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 plus d'informations, consultez 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 plus d'informations, consultez Rôle IAM d'exécuti on 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 plus d'informations, consultez 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 plus d'informations, consultez Types de lancement Amazon ECS.	29 novembre 2017
Changement de nom pour Amazon ECS	Nouveau nom pour Amazon Elastic Container Service (anciennement Amazon EC2 Container Service).	21 novembre 2017

Modification	Description	Date
Mise en réseau des tâches	Les fonctions de mise en réseau des tâches fournies par le mode réseau attribuent aux tâches Amazon ECS les mêmes propriétés de mise en 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 procure plus de contrôle sur la façon dont les applications conteneurisées communiquent entre elles et avec les autres services dans vos VPC. Pour plus d'informations, consultez Mise en réseau des tâches pour les tâches sur les instances Amazon EC2.	14 novembre 2017
Fichier de métadonnées de conteneur Amazon ECS	Les conteneurs Amazon ECS peuvent désormais accéder aux métadonnées, comme leur conteneur Docker ou ID d'image, la configuration de mise en réseau ou les ARN Amazon. Pour plus d'informations, consultez 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 plus d'informations, consultez AMI optimisée pour Amazon ECS.	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 plus d'informations, consultez 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 plus d'informations, consultez LinuxParameters.	22 septembre 2017
Prise en charge de Network Load Balancer	Amazon ECS a ajouté la prise en charge des équilibreurs 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 plus d’informations, consultez Exécuter 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'instances de conteneur de parc d'instances Spot dans la console Amazon ECS. Pour plus d’informations, consultez 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. Pour plus d’informations, consultez Informations connexes.	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 plus d'informations, consultez Drainage des instances de conteneur.	24 janvier 2017
Prise en charge de Docker 1.12	Ajout de la prise en charge de Docker 1.12. Pour plus d'informations, consultez AMI optimisée pour Amazon ECS.	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 plus d'informations, consultez 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 plus d'informations, consultez 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 plus d'informations, consultez Planification de 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 plus d'informations, consultez 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 plus d'informations, consultez Utilisation du pilote du journal awslogs.	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 plus d'informations, consultez Répartissez le trafic du service Amazon ECS à l'aide de l'équilibrage de charge.	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 plus d'informations, consultez Rôle IAM de tâche.	13 juillet 2016
Prise en charge de Docker 1.11	Ajout de la prise en charge de Docker 1.11. Pour plus d'informations, consultez AMI optimisée pour Amazon ECS.	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 plus d'informations, consultez Faites évoluer automatiquement votre service Amazon ECS.	18 mai 2016
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 plus d'informations, consultez ListTaskDefinitions.	17 mai 2016

Modification	Description	Date
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 plus d’informations, consultez AMI optimisée pour Amazon ECS.	5 avril 2016
Prise en charge de Docker 1.9	Ajout de la prise en charge de Docker 1.9. Pour plus d’informations, consultez AMI optimisée pour Amazon ECS.	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
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

Modification	Description	Date
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

Glossaire AWS

Pour connaître la terminologie la plus récente d'AWS, consultez le Glossaire AWS dans la Référence Glossaire AWS.

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.